Dostrajanie klasyfikatora FUZZY LOGIC z użyciem wybranej procedury optymalizacji inspirowanej naturą.

Bartłomiej Leśnicki

Mateusz Niepokój

1.Wstęp

Tematem naszego projektu jest „**Dostrajanie klasyfikatora FUZZY LOGIC z użyciem algorytmu „Krill Herd Algorithm”.** W celu realizacji projektu należało stworzyć fis (fuzzy inference system) a następnie za pomocą wybranej procedury optymalizacji inspirowanej naturą ze zbioru https://github.com/fcampelo/EC-Bestiary doprecyzować parametry fis. Jako zbiór danych testowych użyliśmy (IRIS, WINE oraz SEEDS) z zbioru benchmarkowego UCI MLR http://archive.ics.uci.edu/ml/datasets.php. Poszczególne dane testowe zostały podzielone na dwa podzbiory: dane uczące oraz dane testujące. Algorytmem optymalizującym parametry fis to Krill Herd Algorithm. Należy pamiętać że należy on do algorytmów z grupy algorytmów metaheurystycznych więc wynik dla dość dobrych początkowych parametrów fis nie zawsze ulegnie poprawie.

2. Metody optymalizacji

Optymalizacją jest ważną częścią nauki i badania w tym obszarze są stale prowadzone ze względu na fakt, że prawie wszystkie problemy świata rzeczywistego należą do klasy złożonych problemów optymalizacyjnych, które mają charakter NP-trudny. Celem optymalizacji problemów jest znalezienie najlepszej konfiguracji zmiennych problemu, aby zoptymalizować jego funkcję celu. Problemy te są podzielone na podklasy na ograniczone lub nieograniczone, ciągłe lub dyskretne, jedno lub wielokryterialne oraz statyczne lub dynamiczne. Ze względu na trudny charakter tych problemów, w ciągu ostatnich czterech dekad naukowcy wprowadzili kilka inspirowanych naturą technik algorytmicznych w celu rozwiązania szerokiego zakresu problemów optymalizacyjnych. Popularność tych algorytmów wynika z ich możliwości wyszukiwania i metodologii optymalizacji w radzeniu sobie z problemami wysokiej wymiarowości lepiej niż inne metody oparte na rachunku różniczkowym. Zwykle algorytmy te wywodzą się ze zjawisk naturalnych, gdzie gatunki szukają lepszych warunków życia. Można je podzielić na algorytmy lokalne (Local search), algorytmy ewolucyjne(Evolutionary algorithm) i algorytmy oparty na roju(PSO (particle swarm optimization)). Algorytm oparty na wyszukiwaniu lokalnym rozpoczyna się od pojedynczego rozwiązania tymczasowego, które będzie iteracyjnie ulepszane, aż do osiągnięcia punktu stagnacji w tym samym obszarze rozwiązania początkowego np. symulowane wyżarzanie. Algorytmy ewolucyjne zaczyna od zestawu przypadkowych osobników(populacji), które iteracyjnie łączą rozwiązania(krzyżowanie) i podążają za zasadą przetrwania najsilniejszych osobników, aż do osiągnięcia akceptowalnego rozwiązania np. algorytm genetyczny. Ostatnią klasą jest metoda optymalizacji oparta na roju, która zaczyna od zestawu punktów. W każdej iteracji rozwiązania

są zwykle konstruowane na podstawie informacji historycznych uzyskanych przez poprzednie generacje. Przykładami tego typu algorytmów to: Krill Herd, Catfish, Firely algorithm.

Rysunek 1 Optymalizacja, ilustracja symbolizująca działanie optymalizacji.

3.Krill Herd Algorithm

Algorytm Krill Herd (KH) został zaproponowany przez A.H. Gandomi, A.H. Alavi w 2012 do rozwiązywania problemu optymalizacji globalnej. Jest to algorytm inteligencji stadnej, którego działanie jest modifikowane na podstawie zachowań stadnych osobników kryla. W algorytmie KH funkcja celu dla ruchu kryla jest mierzona najkrótszą odległością każdego pojedynczego kryla od pożywienia i największym zagęszczeniem stada. Każdy osobnik w algorytmie KH modyfikuje swoją pozycję w oparciu o: ruch wywołany przez inne osoby, ruch żerowania oraz losową dyfuzję fizyczną. Algorytm KH jest określany jako potężna technika wyszukiwania, ponieważ zawiera strategie eksploracji i eksploatacji oparte odpowiednio na ruchu żerowania i ruchu wywołanym przez inne osoby. Jest to jeden z najbardziej skutecznych zainspirowanych naturą algorytmów do rozwiązywania praktycznych problemów optymalizacyjnych. Wynika to z jego zalet w zakresie prostoty, elastyczności, wydajności obliczeniowej oraz stochastycznego charakteru, który sprawia, że informacja pochodna nie jest niezbędna. Ponadto, jako technika inteligencji roju, łączy w sobie wydajne działania algorytmu ewolucyjnego wykorzystującego w swoich ramach elementy krzyżowania i mutacji, dzięki czemu struktura wyszukiwania jest silniejsza niż inne algorytmy pod względem współczynnika zbieżności. Sukces algorytmu KH został odnotowany w wielu obszarach, takich jak globalne funkcje optymalizacji, optymalizacja sieci, problem ekonomicznej wysyłki, optymalizacja konstrukcji. Czasami jednak może utknąć w niektórych lokalnych optimach, a zatem nie może być w stanie w pełni wdrożyć globalnego wyszukiwania. Te niedociągnięcia doprowadziły do jego modyfikacji pod względem koncepcji

i hybrydyzacji z komponentami z innych metaheurystyk, gdy są wykorzystywane do rozwiązywania problemów wysokiej wymiarowości.

Rysunek 2 Ławica Krylu

Rysunek 3 Pojedynczy osobnik Krylu

4. Fuzzy Logic

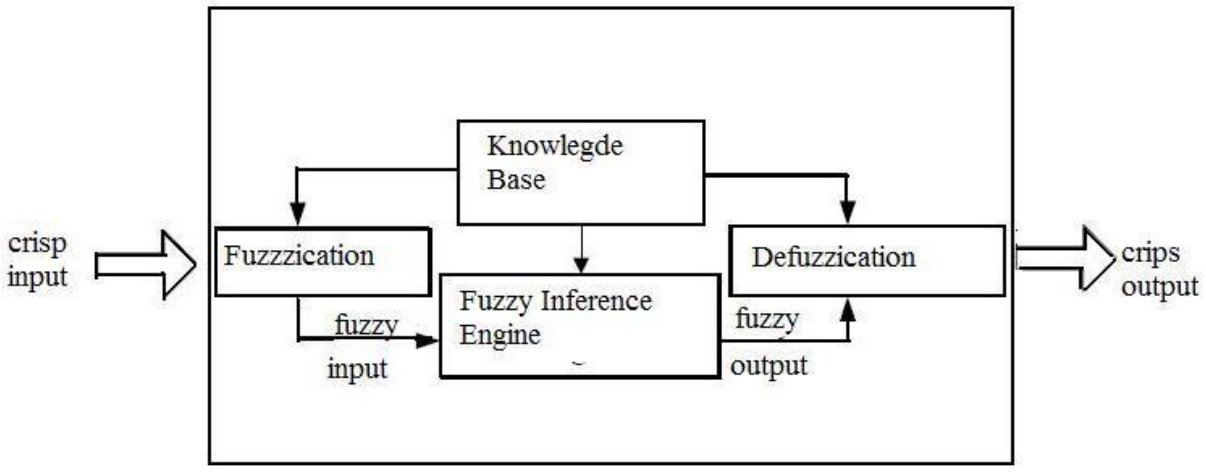
Logika rozmyta – jedna z logik wielowartościowych, stanowi uogólnienie klasycznej dwuwartościowej logiki. Została zaproponowana przez Lotfi Zadeha, jest ściśle powiązana z jego teorią zbiorów rozmytych. W logice rozmytej między stanem 0 (fałsz) a stanem 1 (prawda) rozciąga się szereg wartości pośrednich, które określają stopień przynależności elementu do zbioru.

Logika rozmyta okazała się bardzo przydatna w zastosowaniach inżynierskich, gdzie klasyczna logika klasyfikująca jedynie według kryterium prawda/fałsz nie potrafi skutecznie poradzić sobie z wieloma niejednoznacznościami i sprzecznościami. Znajduje wiele zastosowań, między innymi w elektronicznych systemach sterowania (maszynami, pojazdami i automatami), zadaniach eksploracji danych czy też w budowie systemów ekspertowych.

Metody logiki rozmytej wraz z algorytmami ewolucyjnymi i sieciami neuronowymi stanowią nowoczesne narzędzia do budowy inteligentnych systemów mających zdolności uogólniania wiedzy.

Wnioskowanie rozmyte to metoda, która interpretuje wartości w wektorze wejściowym i na podstawie pewnych zestawów reguł przypisuje wartości do wektora wyjściowego. W logice rozmytej prawdziwość każdego stwierdzenia staje się kwestią pewnego stopnia.

Wnioskowanie rozmyte to proces formułowania odwzorowania danych wejściowych na dane wyjściowe przy użyciu logiki rozmytej. Mapowanie zapewnia następnie podstawę, na podstawie której można podejmować decyzje lub rozpoznawać wzorce. Proces wnioskowania rozmytego obejmuje: funkcje przynależności, operatory logiki rozmytej i reguły if-then. Można zaimplementować dwa główne typy systemów wnioskowania rozmytego: typu Mamdaniego (1977) i typu Sugeno (1985). Te dwa typy systemów wnioskowania różnią się sposobem określania wyników.



Rysunek 4 Schemat systemu wnioskowania rozmytego

5. Realizacja projektu

Projekt został napisany w języku matlab z użyciem Fuzzy inference system. Jako algorytm do optymalizacji parametrów FIS został użyty Krill Herd. Kolejne kroki projektu:

1. Inicjalizacja zbiorów testowych IRIS, SEEDS, WINE z zbioru benchmarkowego UCI MLR http://archive.ics.uci.edu/ml/datasets.php.

2. Randomizujemy próbki zbiorów i dzielimy je na dwa podzbiory dane uczące(80%) oraz dane testujące(20%). Metoda CV-5.

3. Inicjujemy FIS (fuzzy inference system) za pomocą ręcznie modelowanych zasad osobno dla każdego zbioru danych. Odczytujemy FIS funkcją readfis.

```
fis = readfis('zbiór_danych.fis');
```

Dla:
-IRIS zbiór testowy ma 4 cechy więc nasz fis ma 4 wejścia.

-SEED zbiór testowy ma 7 cech, fis 7 wejścia.

-WINE wybraliśmy 8 parametrów opisujący zbiór oraz 175 danych wejściowych zamiast 178.

4. Ewaluujemy FIS funkcją evalfis , która zwraca jednowyjściowy Sugeno fis.

```
y_out = evalfis(fis, x_train);
y_test = evalfis(fis, x_testing);
```

5. Z FIS pobieramy parametry do optymalizacji algorytmem Krill Herd

```
[in, out] = getTunableSettings(fis);
paramVals = getTunableValues(fis, [in; out]);
```

6. Definiujemy granice danych wejścowych jako LB i UB.

7. Wywołujemy algorytm KH

```
x = KH(fis, LB, UB, @cost, x_train, y_train);
```

8. Tworzymy referencyjny FIS metodą genfis.

```
fis_SC=genfis(x_u,y_u,genfisOptions('SubtractiveClustering'))
```

Ten sposób generacji fis daje bardzo dobre wyniki klasyfikacji danych

9. Analiza wyników

6. Analiza Wyników:

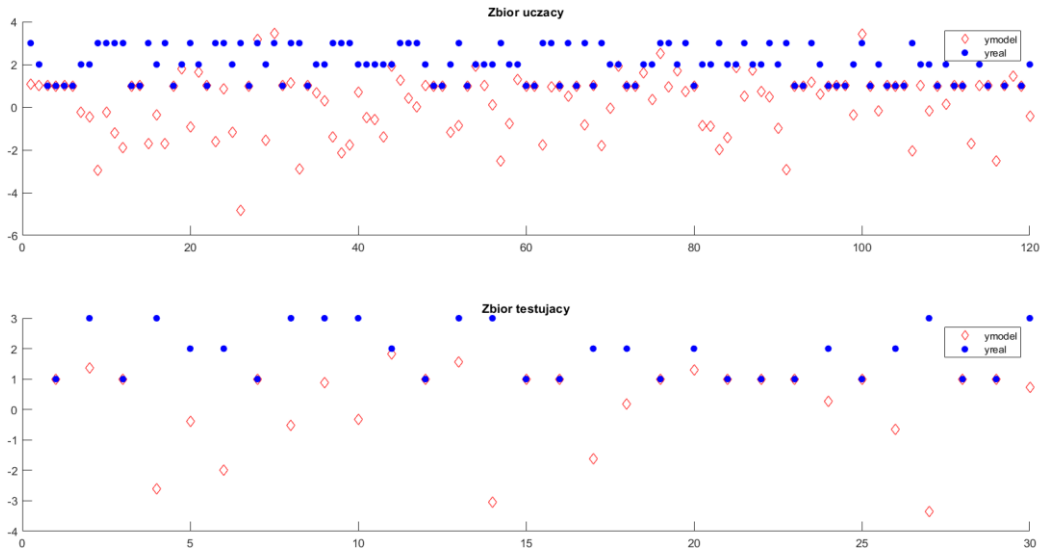
6.1 IRIS

Zbiór testowy IRIS reprezentuje kwiaty irysa posiadające 4 cechy, które są 4 wejściami dla FIS. Kwiaty dzielone są na 3 gatunki.

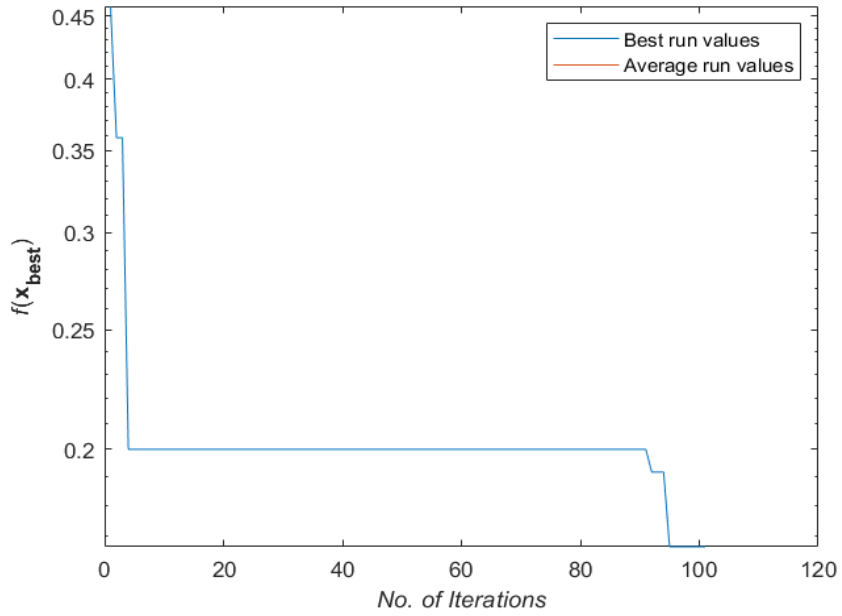
Dla manualnie wygenerowanego FIS uzyskujemy wartości poprawnej klasyfikacji rzędu 43%. Natomiast dla algorytmu Krill Herd widać znaczącą poprawę jakości klasyfikacji. W przedstawionych poniżej wynikach, parametry algorytmu KH to 100 iteracji oraz 40 osobników krylu. Dla zadanych parametrów klasyfikacja wynosi średnio 73%. Można zauważyć że dla większej liczby iteracji oraz krylu zwiększa się procentowo poprawność klasyfikacji, zwiększa się również czas wykonania programu. Odchylenie standardowe poprawności klasyfikacji poszczególnych iteracji programu dla zadanych parametrów 15.144% co świadczy o heurystycznym charakterze algorytmu. Dla manualnego FIS w tym przypadku odchylenie standardowe wynosi 8.794%.

```
Iteracja: 1
Percentage of corectly qualified cases (manual FIS) - teaching set: 42%
Percentage of corectly qualified cases (manual FIS) - testing set: 43%
Percentage of corectly qualified cases (KH) - teaching set:  72%
Percentage of corectly qualified cases (KH) - testing set:  53%
Iteracja: 2
Percentage of corectly qualified cases (manual FIS) - teaching set: 47%
Percentage of corectly qualified cases (manual FIS) - testing set: 27%
Percentage of corectly qualified cases (KH) - teaching set:  71%
Percentage of corectly qualified cases (KH) - testing set:  53%
Iteracja: 3
Percentage of corectly qualified cases (manual FIS) - teaching set: 42%
Percentage of corectly qualified cases (manual FIS) - testing set: 43%
Percentage of corectly qualified cases (KH) - teaching set:  73%
Percentage of corectly qualified cases (KH) - testing set:  80%
Iteracja: 4
Percentage of corectly qualified cases (manual FIS) - teaching set: 40%
Percentage of corectly qualified cases (manual FIS) - testing set: 53%
Percentage of corectly qualified cases (KH) - teaching set:  82%
Percentage of corectly qualified cases (KH) - testing set:  80%
Iteracja: 5
Percentage of corectly qualified cases (manual FIS) - teaching set: 42%
Percentage of corectly qualified cases (manual FIS) - testing set: 47%
Percentage of corectly qualified cases (KH) - teaching set:  83%
Percentage of corectly qualified cases (KH) - testing set:  90%
```

Rysunek 5 Porównanie poprawności klasyfikacji dla manualnie wygenerowanego FIS oraz dla jego wersji z poprawionymi parametrami KH. Zbiór IRIS



Rysunek 6 Klasyfikacja danych zbiór IRIS



Rysunek 7 Minimalizacja funkcji kosztu algorytmem Krill Herd

Dla 20 osobników krylu oraz 20 iteracji KH otrzymujemy średnią wartość prawidłowej klasyfikacji rzędu 61% oraz odchylenie standardowe 8.055%.

```
Iteracja: 1
Percentage of corectly qualified cases (manual FIS) - teaching set: 45%
Percentage of corectly qualified cases (manual FIS) - testing set: 33%
Percentage of corectly qualified cases (KH) - teaching set:  57%
Percentage of corectly qualified cases (KH) - testing set:  47%
Iteracja: 2
Percentage of corectly qualified cases (manual FIS) - teaching set: 44%
Percentage of corectly qualified cases (manual FIS) - testing set: 37%
Percentage of corectly qualified cases (KH) - teaching set:  76%
Percentage of corectly qualified cases (KH) - testing set:  67%
Iteracja: 3
Percentage of corectly qualified cases (manual FIS) - teaching set: 42%
Percentage of corectly qualified cases (manual FIS) - testing set: 43%
Percentage of corectly qualified cases (KH) - teaching set:  61%
Percentage of corectly qualified cases (KH) - testing set:  70%
Iteracja: 4
Percentage of corectly qualified cases (manual FIS) - teaching set: 40%
Percentage of corectly qualified cases (manual FIS) - testing set: 53%
Percentage of corectly qualified cases (KH) - teaching set:  68%
Percentage of corectly qualified cases (KH) - testing set:  60%
Iteracja: 5
Percentage of corectly qualified cases (manual FIS) - teaching set: 42%
Percentage of corectly qualified cases (manual FIS) - testing set: 47%
Percentage of corectly qualified cases (KH) - teaching set:  67%
Percentage of corectly qualified cases (KH) - testing set:  63%
```

Rysunek 8 Porównanie poprawności klasyfikacji dla manualnie wygenerowanego FIS oraz dla jego wersji z poprawionymi parametrami KH. Zbiór IRIS mniejsza ilość iteracji i osobników Krylu niż w przypadku przedstawionym na rysunku 5

Dla automatycznie wygenerowanego FIS z opcją SubtractiveClustering, wyniki są znacząco lepsze:

```
Iteracja: 1
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.167%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 96.667%
Iteracja: 2
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.167%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 96.667%
Iteracja: 3
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.167%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 96.667%
Iteracja: 4
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.167%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 96.667%
Iteracja: 5
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.167%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 96.667%
```

Rysunek 9 Procentowe wartości klasyfikacji automatycznie wygenerowanym FIS – Zbiór IRIS

Średnia wartość dobrze zaklasyfikowanych przypadków wynosi 96.667% z odchyleniem standardowym 0. Świadczy to o dobrze zoptymalizowanych funkcjach matlaba. Algorytm jest

na tyle dobry że poprawiając nasz manualny FIS algorytmem KH nie osiągnęlibyśmy tak prawidłowych wyników przy rozsądnym czasie wykonania programu.

6.2 SEEDS

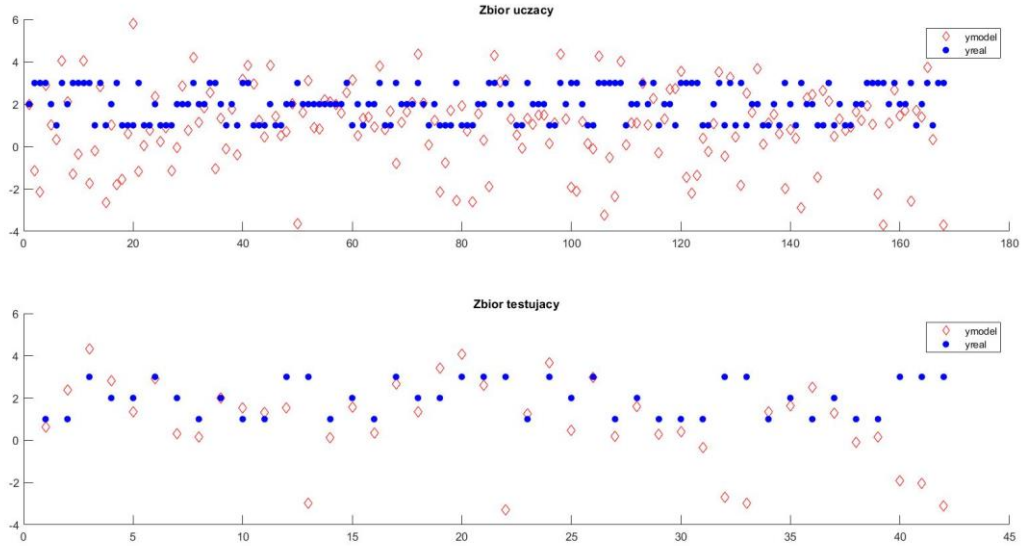
Zbiór testowy SEEDS składa się z 7 cech służących jako 7 wejść do FIS. W ramach zbioru dostępne są 3 gatunki do klasyfikacji.

Dla manualnie wygenerowanego FIS uzyskujemy wartości poprawnej klasyfikacji rzędu 27%. Dla zmodyfikowanych parametrów algorytmem Krill Herd widać poprawę jakości klasyfikacji. Parametry algorytmu KH to 100 iteracji oraz 20 osobników krylu. Dla zadanych parametrów klasyfikacja wynosi średnio 40%. Można zauważyć że tak jak w IRIS dla zbioru SEEDS większa liczba iteracji oraz krylu zwiększa procentowo poprawność klasyfikacji, zwiększa się również czas wykonania programu w tym przypadku znacznie bardziej niż IRIS ze względu na większą ilość cech do modyfikacji. Odchylenie standardowe poprawności klasyfikacji poszczególnych iteracji programu dla zadanych parametrów 18.319%. Tak duża wartość odchylenia standardowego wynika z metahurystycznej natury algorytmu KH. Widać to na przykładzie poniższych wyników gdzie poprawna ilość klasyfikacji dla jednej z iteracji wykonania programu uzyskujemy wartość dużo gorszą niż przed modyfikacją-12% a dla następnej wartość znacznie lepszą-64%. Dla manualnego FIS w tym przypadku odchylenie standardowe wynosi 9.943%

```
Iteracja: 1
Percentage of corectly qualified cases (manual FIS) - teaching set: 27%
Percentage of corectly qualified cases (manual FIS) - testing set: 29%
Percentage of corectly qualified cases (KH) - teaching set:  30%
Percentage of corectly qualified cases (KH) - testing set:  31%
Iteracja: 2
Percentage of corectly qualified cases (manual FIS) - teaching set: 24%
Percentage of corectly qualified cases (manual FIS) - testing set: 40%
Percentage of corectly qualified cases (KH) - teaching set:  38%
Percentage of corectly qualified cases (KH) - testing set:  12%
Iteracja: 3
Percentage of corectly qualified cases (manual FIS) - teaching set: 32%
Percentage of corectly qualified cases (manual FIS) - testing set: 10%
Percentage of corectly qualified cases (KH) - teaching set:  52%
Percentage of corectly qualified cases (KH) - testing set:  55%
Iteracja: 4
Percentage of corectly qualified cases (manual FIS) - teaching set: 27%
Percentage of corectly qualified cases (manual FIS) - testing set: 29%
Percentage of corectly qualified cases (KH) - teaching set:  53%
Percentage of corectly qualified cases (KH) - testing set:  64%
Iteracja: 5
Percentage of corectly qualified cases (manual FIS) - teaching set: 27%
Percentage of corectly qualified cases (manual FIS) - testing set: 29%
Percentage of corectly qualified cases (KH) - teaching set:  54%
Percentage of corectly qualified cases (KH) - testing set:  40%
```

Rysunek 10 Porównanie poprawności klasyfikacji dla manualnie wygenerowanego FIS oraz dla jego wersji z poprawionymi parametrami KH. Zbiór SEEDS

.



Rysunek 11 Klasyfikacja danych zbiór SEEDS.

Również w tym przypadku automatycznie wygenerowany FIS z opcją SubtractiveClustering, zwraca wyniki znacząco lepsze:

```
Iteracja: 1
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set uczący: 100.000%
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set testujący: 85.714%
Iteracja: 2
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set uczący: 100.000%
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set testujący: 85.714%
Iteracja: 3
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set uczący: 100.000%
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set testujący: 85.714%
Iteracja: 4
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set uczący: 100.000%
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set testujący: 85.714%
Iteracja: 5
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set uczący: 100.000%
Procent dobrze zkwalifikowanych przypadków (SubtractiveClustering FIS) - set testujący: 85.714%

Glowna_srednia_SC =

  85.714285700000005


odchylenie_standardowe_SC =

     0
```

Rysunek 12 Procentowe wartości klasyfikacji automatycznie wygenerowanym FIS – Zbiór SEEDS

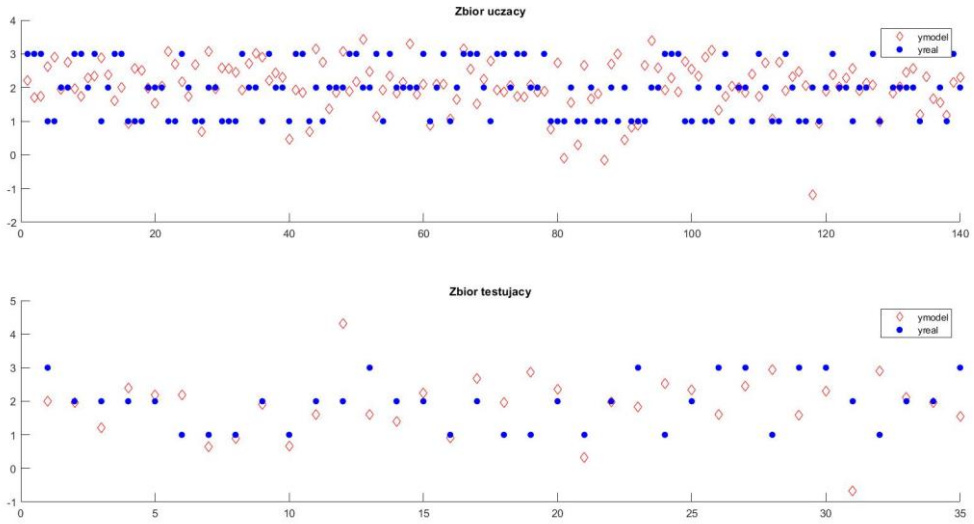
Dla zbioru danych testowych SEEDS automatycznie wygenerowany FIS poprawnie klasyfikuje dane na poziomie 85,714%. Jest to wynik o około 11 punktów procentowych gorszy niż dla zbioru IRIS. Jednak wynik ten jest wystarczająco dobry a czas generacji FIS jest bardzo mały rzędu jednej sekundy. Na tym samym środowisku dla proponowanego rozwiązania

optymalizacji parametrów ręcznie wygenerowanego FIS z podanymi wyżej parametrami algorytmu KH czas liczony jest w godzinach a wynik znacząco gorszy. Wynika to z kosztownej funkcji obliczania kosztu, gdzie pobierane i zapisywane są dane z FIS.

6.3 WINE

Dla zbioru testowego WINE wybraliśmy 8 parametrów opisujący zbiór oraz 175 danych wejściowych zamiast 178. W ramach zbioru dostępne są 3 gatunki do klasyfikacji.

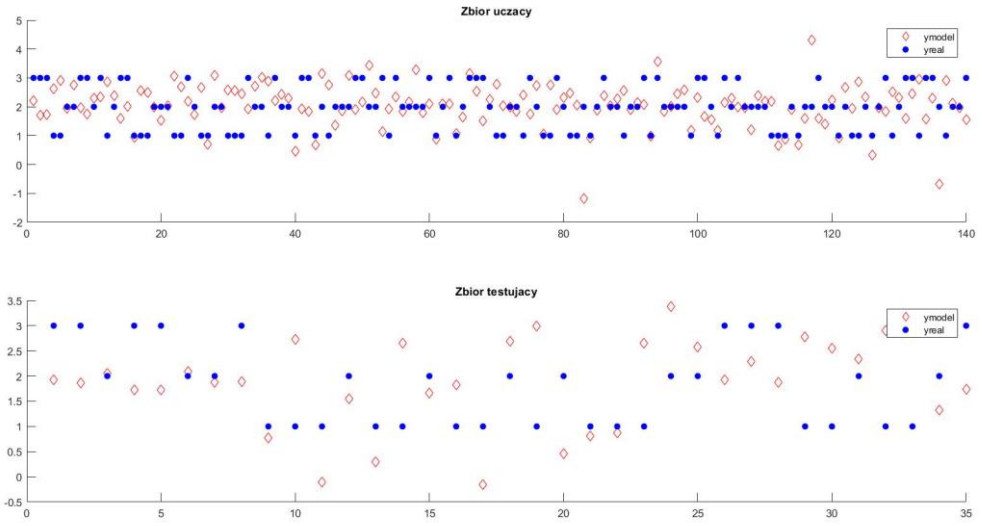
Dla manualnie wygenerowanego FIS uzyskujemy wartości poprawnej klasyfikacji rzędu 38%. Natomiast dla modyfikacji parametrów algorytmem Krill Herd widać poprawę jakości klasyfikacji. W przedstawionych poniżej wynikach, parametry algorytmu KH to 40 iteracji oraz 25 osobników krylu. Dla zadanych parametrów klasyfikacja wynosi średnio 62%. Można zauważyć że dla większej liczby iteracji oraz krylu zwiększa się procentowo poprawność klasyfikacji, zwiększa się również czas wykonania programu w tym przypadku znacznie bardziej niż IRIS ze względu na większą ilość cech do modyfikacji. Odchylenie standardowe poprawności klasyfikacji poszczególnych iteracji programu dla zadanych parametrów 2.555%. Dla manualnego FIS w tym przypadku odchylenie standardowe wynosi 9.664%.



Rysunek 13 Klasyfikacja danych zbiór WINE

```
Iteracja: 1
Percentage of corectly qualified cases (manual FIS) - teaching set: 41%
Percentage of corectly qualified cases (manual FIS) - testing set: 29%
Percentage of corectly qualified cases (KH) - teaching set:  68%
Percentage of corectly qualified cases (KH) - testing set:  63%
Iteracja: 2
Percentage of corectly qualified cases (manual FIS) - teaching set: 37%
Percentage of corectly qualified cases (manual FIS) - testing set: 43%
Percentage of corectly qualified cases (KH) - teaching set:  55%
Percentage of corectly qualified cases (KH) - testing set:  60%
Iteracja: 3
Percentage of corectly qualified cases (manual FIS) - teaching set: 41%
Percentage of corectly qualified cases (manual FIS) - testing set: 46%
Percentage of corectly qualified cases (KH) - teaching set:  66%
Percentage of corectly qualified cases (KH) - testing set:  66%
Iteracja: 4
Percentage of corectly qualified cases (manual FIS) - teaching set: 35%
Percentage of corectly qualified cases (manual FIS) - testing set: 51%
Percentage of corectly qualified cases (KH) - teaching set:  61%
Percentage of corectly qualified cases (KH) - testing set:  60%
Iteracja: 5
Percentage of corectly qualified cases (manual FIS) - teaching set: 37%
Percentage of corectly qualified cases (manual FIS) - testing set: 43%
Percentage of corectly qualified cases (KH) - teaching set:  69%
Percentage of corectly qualified cases (KH) - testing set:  66%
```

Rysunek 14 Porównanie poprawności klasyfikacji dla manualnie wygenerowanego FIS oraz dla jego wersji z poprawionymi parametrami KH. Zbiór WINE



Rysunek 15 Klasyfikacja danych zbiór WINE wersja 2

Dla automatycznie wygenerowanego FIS z opcją SubtractiveClustering, wyniki są znacząco lepsze:

```
Iteracja: 1
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.286%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 88.571%
Iteracja: 2
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.286%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 88.571%
Iteracja: 3
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.286%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 88.571%
Iteracja: 4
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.286%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 88.571%
Iteracja: 5
Percentage of corectly qualified cases (SubtractiveClustering FIS) -teaching set: 99.286%
Percentage of corectly qualified cases (SubtractiveClustering FIS) -test set: 88.571%
```

Rysunek 16 Procentowe wartości klasyfikacji automatycznie wygenerowanym FIS – Zbiór WINE

Średnia wartość dobrze zaklasyfikowanych przypadków dla kilku uruchomień programu wacha się między 80% a 95%. Również w tym przypadku algorytm matlaba wykazuję supremację względem naszych prób dostosowania parametrów FIS algorytmem Krill Herd.

7. Podsumowanie i Wnioski

Metody sztucznej inteligencji są jedną z najbardziej rozwijających się dziedzin nauki. Systemy takie jak fuzzy inference system oparte na logice rozmytej pozwalają w prosty sposób sklasyfikować dane. Odpowiedni dobór reguł często automatyczny, umożliwia klasyfikację zarówno prostych zbiorów testowych opartych na ścisłych parametrach, po przez skomplikowane wielowymiarowe dane, aż po rozpoznawanie obrazów czy mowy. Sztuczna inteligencja pomaga w procesach optymalizacji. Ma to niebagatelny wpływ na rozwój ludzkości. Optymalizacja odpadów, minimalizacja zużycia energii czy kosztów produkcji oddziałuje bezpośrednio na naszą przyszłość. Odpowiednie algorytmy metaheurystyczne potrafią rozwiązać wiele problemów klasy NP-trudnych nieosiągalnych tradycyjnymi metodami.

W realizowanym przez nas projekcie, poruszyliśmy temat zarówno systemu wnioskowania rozmytego FIS jak i algorytmów metaheurystycznych - KH (Krill Herd algorithm). Dowiedliśmy że w za pomocą KH można polepszyć parametry FIS w celu lepszej klasyfikacji danych. Przedstawione wyniki wyraźnie pokazują wzrost procentowy poprawnie rozpoznanych informacji. Poprawność klasyfikacji rosła wraz z zwiększaniem liczby iteracji algorytmu KH oraz liczby osobników krylu.

W trakcie realizacji projektu napotkaliśmy problem związany w wydajnością naszego programu. Algorytm Krill Herd dla dużej liczby iteracji liczył się bardzo długo. Nie wynikało to bezpośrednio z jego implementacji lecz z funkcji obliczającej koszt. W ciele tej funkcji są pobierane i zmieniane dane w systemie wnioskowania rozmytego. Te operacje są dość kosztowne i dla bardzo wielu iteracji algorytm dział zdecydowanie zbyt wolno. Dodatkowo aby

poprawić działanie algorytmu należy odpowiednio zwiększyć jego parametry (liczbę iteracji i kryli) co przy tak kosztownej funkcji obliczania kosztu znacznie wydłuża działanie algorytmu.

Problem świadczy o tym że wybrana metodyka zmiany parametrów nie jest najlepszą opcją. W projekcie wykorzystaliśmy również automatyczny sposób generowania FIS funkcją genfis. Tak wygenerowany system dużo lepiej klasyfikował dane, a jego stworzenie nie wymagało długotrwałych obliczeń.

Podsumowując projekt wykazał że zarówno logika rozmyta jak i algorytmy metaheurystczne inspirowane naturą to potężne narzędzia informatyczne. Można dzięki nimi w dobry sposób sklasyfikować dane. Najważniejsze jest jednak poprawne użycie tych narzędzi jak i poprawne zadanie problemu. W niektórych przypadkach istnieją łatwiejsze i bardziej wydajne metody rozwiązywania danych zagadnień a użycie sztucznej inteligencji nie zawsze jest najlepszym wyborem.

8. Źródła:

https://en.wikipedia.org/wiki/Adaptive_neuro_fuzzy_inference_system

https://www.sciencedirect.com/topics/engineering/fuzzy-inference

https://www.sciencedirect.com/science/article/pii/S1568494616304355#bib0085

https://pl.wikipedia.org/wiki/Logika_rozmyta

https://www.mathworks.com/matlabcentral/fileexchange/55486-krill-herd-algorithm

https://www.mathworks.com/products/fuzzy-logic.html

Materiały dostępne w ramach realizacji przedmiotu Metody Inteligencji Obliczeniowe