WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI
POLITECHNIKI RZESZOWSKIEJ

Sztuczna inteligencja
Laboratorium

Synteza układu wnioskującego

Stanislau Antanovich
nr. indeksu: 173590
gr. lab: L04

Spis treści

1 Wstęp **2**
1.1 Cel ćwiczenia 2

2 System ekspertowy typu Mamdaniego **2**
2.1 Problem 2
2.2 Realizacja 3
2.2.1 Inicjowanie modułów 3
2.2.2 Tworzenie zmiennych stanu poprzednika "obsługa" oraz następnika "napiwek" 3
2.2.3 Dodanie zbiorów rozmytych 3
2.2.4 Podgląd zbiorów rozmytych 4
2.2.5 Definicja reguł 4
2.2.6 Dodanie reguł do systemu rozmytego 4
2.2.7 Sprawdzenie działania systemu 4
2.2.8 Sprawdzenie działania systemu dla wartości obsługi . . . 5
2.2.9 Dodanie drugiej wejściowej 6
2.2.10 Dodanie reguł 4 i 5 6
2.2.11 Sprawdzenie działania systemu dla wartości "obsługi" . . 6
2.2.12 Sprawdzenie działania systemu dla wartości "obsługi" i "jedzenia" 6

3 Modyfikacja systemu **7**

4 Przykładowe zadania zaliczeniowe **7**

5 Wnioski **7**

Spis rysunków

1 *Obsługa* 2
2 *Napiwek* 3
3 *Wyostrzenie metodą środka ciężkości dla wejścia obsługa = 0* . . 5
4 *Wyostrzenie metodą środka ciężkości dla wejścia obsługa = 10* . 5
5 *Powierzchnia prejścia systemu "napiwek" o jednym wejściu* . . . 6
6 *Powierzchnia przejścia systemu "napiwek" o dwóch wejściach* . . 6

1 Wstęp

1.1 Cel ćwiczenia

Laboratorium składa się z trzech zasadniczych części. Część 2 ma na celu zapoznanie się ze sposobem syntezy rozmytego systemu ekspertowego typu Mamdaniego z wykorzystaniem biblioteki **scikit-fuzzy**. W części **??**, należy zapoznać się z ideą działania systemu Mamdaniego a następnie dokonać modyfikacji systemu wykonanego w części **??**. Część **??** laboratorium polega na wykonaniu przykładowego zadania zaliczeniowego.

2 System ekspertowy typu Mamdaniego

2.1 Problem

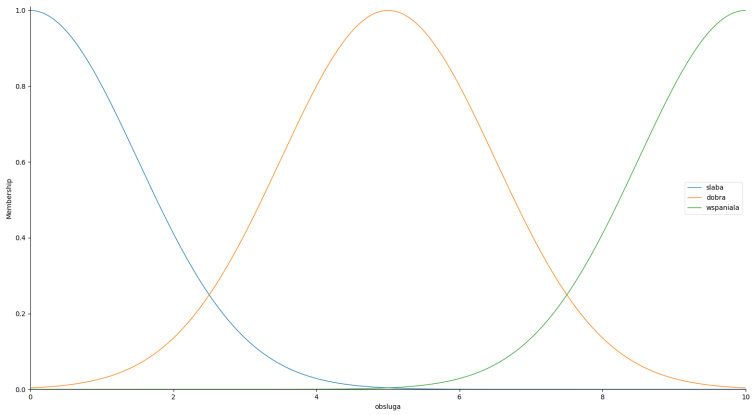
Zaprojektować rozmyty układ ekspertowy doradzający ile napiwku pozostawić w restauracji na podstawie oceny jakości obsługi oraz jakości jedzenia. Jakość obsługi i jakość jedzenia będzie oceniana w skali od 1 do 10, gdzie 10 reprezentuję ocenę maksymalną, natomiast napiwek będzie liczbą z przedziału [0,30] reprezentującą procent wartości rachunku.

Baza reguł będzie składała się z 5 reguł. System zostanie wykonany w dwóch etapach. W **etapie pierwszym**(rys. 1 i 2) system będzie zbudowany z jednego wejścia(*obsługa*) i jednego wyjścia(*napiwek*) oraz 3 reguł postaci:

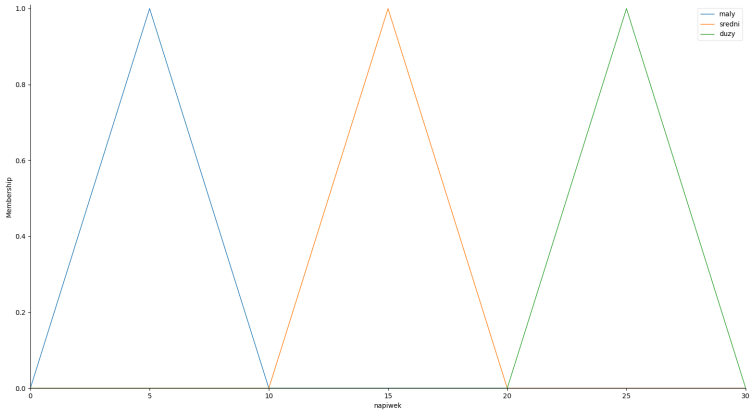
R1 *jeżeli obsługa jest słaba, to napiwek jest mały*

R2 *jeżeli obsługa jest dobra, to napiwek jest średni*

R3 *jeżeli obsługa jest wspaniała, to napiwek jest duży*



Rysunek 1: *Obsługa*



Rysunek 2: *Napiwek*

W **etapie drugim** do systemu zostanie dodane gruga zmienna wejściowa *jedzenie*(rys. **??**) oraz 2 dodatkowe reguły

R4 *jeżeli jedzenie jest zepsute, to napiwek jest mały*

R5 *jeżeli jedzenie jest wyborne, to napiwek jest duży*

2.2 Realizacja

2.2.1 Inicjowanie modułów

W tej sekcji inicjujemy wszystkie wymagane biblioteki dla prawidłowego działania programu.

```
1 import numpy as np
2 import skfuzzy as fuzz
3 from skfuzzy import control as ctrl
4 import matplotlib.pyplot as plt
```

2.2.2 Tworzenie zmiennych stanu poprzednika “obsługa” oraz następnika “napiwek”

W tej sekcji tworzymy zmienne stanu poprzednika “obsługa” oraz następnika “napiwek”.

```
1 obsluga = ctrl.Antecedent(np.arange(0, 10.01, 0.01), 'obsluga')
2 napiwek = ctrl.Consequent(np.arange(0, 30.01, 0.01), 'napiwek')
```

2.2.3 Dodanie zbiorów rozmytych

W tej sekcji do zminnej *obsługa* dodajemy następujące zbiory: *slaba, dobra, wspaniala.*

Zbiór *slaba* o centrum umieszczonym w punkcie uniwersum równym **0** i rozpiętości wynoszącej **1.5**. Dla zbiorów *dobra* i *wspaniala* o centrach ulokowanych w punktach odpowiednio **5** oraz **10** i rozpiętości wynoszącej **1.5**

Dla zmiennej *napiwek* dodajemy zbiory trójkątne: *maly, sredni, duzy* o parametrach [0, 5, 10], [10, 15, 20] i [20, 25, 30] odpowiednio.

```

# Define membership functions for 'obsluga'
obsluga['slaba'] = fuzz.gaussmf(obsluga.universe, 0, 1.5)
obsluga['dobra'] = fuzz.gaussmf(obsluga.universe, 5, 1.5)
obsluga['wspaniala'] = fuzz.gaussmf(obsluga.universe, 10, 1.5)

# Define membership functions for 'napiwek'
```

2.2.4 Podgląd zbiorów rozmytych

Podgląd zbiorów rozmytch można zrealizować metodą `view()` dla poszczególnych zmiennych stanu.

```
napiwek['sredni'] = fuzz.trimf(napiwek.universe, [10, 15, 20])
napiwek['duzy'] = fuzz.trimf(napiwek.universe, [20, 25, 30])
```

2.2.5 Definicja reguł

W tej sekcji definiujemy reguły.

```
# Define membership functions for 'jedzenie'
jedzenie['zepsute'] = fuzz.trapmf(jedzenie.universe, [-2, 0, 1, 3])
jedzenie['wyborne'] = fuzz.trapmf(jedzenie.universe, [7, 9, 10, 12])
```

2.2.6 Dodanie reguł do systemu rozmytego

W tej sekcji dodajemy powyżej zdefiniowane reguły do systemu rozmytego.

```
# View membership functions
obsluga.view()
```

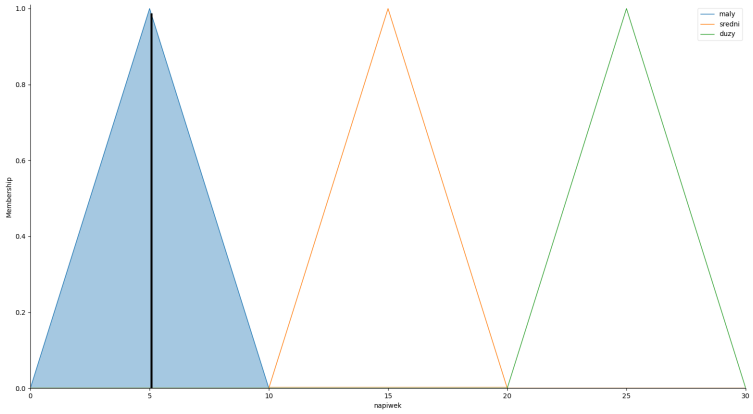
2.2.7 Sprawdzenie działania systemu

W tej sekcji sprawdzamy działanie systemu dla wartości obsługi równej **0**(rys. 3) oraz dla wartości równej **10**(rys. 4).

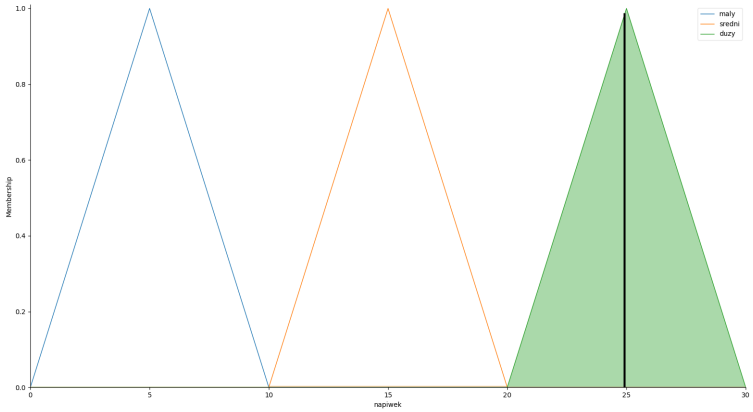
```
jedzenie.view()

# Define rules
regula1 = ctrl.Rule(obsluga['slaba'], napiwek['maly'])
regula2 = ctrl.Rule(obsluga['dobra'], napiwek['sredni'])
```

Wartość napiwku dla wartości obsługi wynosi: 5.07657801
Wartość napiwku dla wartości obsługi wynosi: 24.9234219



Rysunek 3: *Wyostrzenie metodą środka ciężkości dla wejścia obsługa = 0*



Rysunek 4: *Wyostrzenie metodą środka ciężkości dla wejścia obsługa = 10*

2.2.8 Sprawdzenie działania systemu dla wartości obsługi

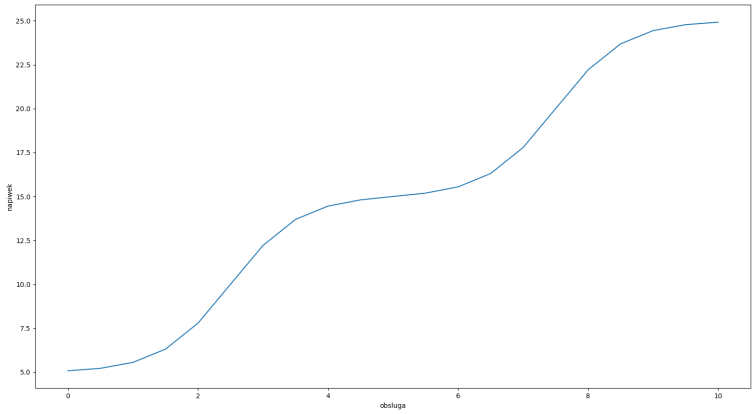
W tej sekcji sprawdzamy działanie systemu dla wartości obsługi od 0 do 10(rys. 5 wykres funkcji napiwek=f(obsluga)).

```
regula4 = ctrl.Rule(jedzenie['zepsute'], napiwek['maly'])
regula5 = ctrl.Rule(jedzenie['wyborne'], napiwek['duzy'])

# Set up control system
napiwek_ctr = ctrl.ControlSystem([regula1, regula2, regula3, regula4,
    regula5])
napiwek_sym = ctrl.ControlSystemSimulation(napiwek_ctr)

# Input values and compute
napiwek_sym.input['obsluga'] = 0
napiwek_sym.input['jedzenie'] = 0
```

```
napiwek_sym.compute()
print('Wynik', napiwek_sym.output['napiwek'])
napiwek.view(sim=napiwek_sym)
```



Rysunek 5: *Powierzchnia prejścia systemu "napiwek" o jednym wejściu*

2.2.9 Dodanie drugiej wejściowej

W tej selcji dodajemy drugą wejściową zmienną stanu "jedzenie" o trapezoidalnych(`trapmf`) zbiorach rozmytych "zepsute" oraz "wyborne".

```
napiwek['sredni'] = fuzz.trimf(napiwek.universe, [10, 15, 20])
napiwek['duzy'] = fuzz.trimf(napiwek.universe, [20, 25, 30])
```

2.2.10 Dodanie reguł 4 i 5

W tej sekcji dodajemy reguły 4 i 5 analogicznie do punktu **??**.

2.2.11 Sprawdzenie działania systemu dla wartości "obsługi"

W tej sekcji sprawdzamy działanie systemu dla wartości obsługi równej **0** oraz wartości jedzenia równej **0**.

2.2.12 Sprawdzenie działania systemu dla wartości "obsługi" i "jedzenia"

W tej sekcji sprawdzamy działanie systemu dla wartości obsługi i jedzenia od 0 do 10(rys. 6)

`Figure_6.png`

Rysunek 6: *Powierzchnia przejścia systemu "napiwek" o dwóch wejściach*

3 Modyfikacja systemu

4 Przykładowe zadania zaliczeniowe

5 Wnioski