Міністерство освіти та науки України
Львівський національний університет імені Івана Франка

Факультет електроніки та
Комп’ютерних технологій

Звіт

Про виконання лабораторної роботи №2
“Побудова функцій приналежності нечіткої множини на основі попарних порівнянь”

Виконав:
Студент групи ФеІ-44
Сапанюк М.І.
Перевірила:
Притула М.

Львів 2022

Мета:

Ознайомитися з непрямим методом побудови функцій приналежності нечіткої множини на основі попарних порівнянь елементів нечіткої множини експертом.

Хід роботи:

1. На основі опитування експерта побудувати матрицю попарних порівнянь елементів нечіткої множини «висока людина», які відповідають росту: 170, 175, 180, 185, 190, 195, 200 см, і нечіткої множини «низька людина»: 150, 155, 160, 165, 170, 175, 180 см. Зокрема, для кожної пари елементів універсальної множини X оцінити перевагу одного елемента над іншим відносно певної властивості нечіткої множини.

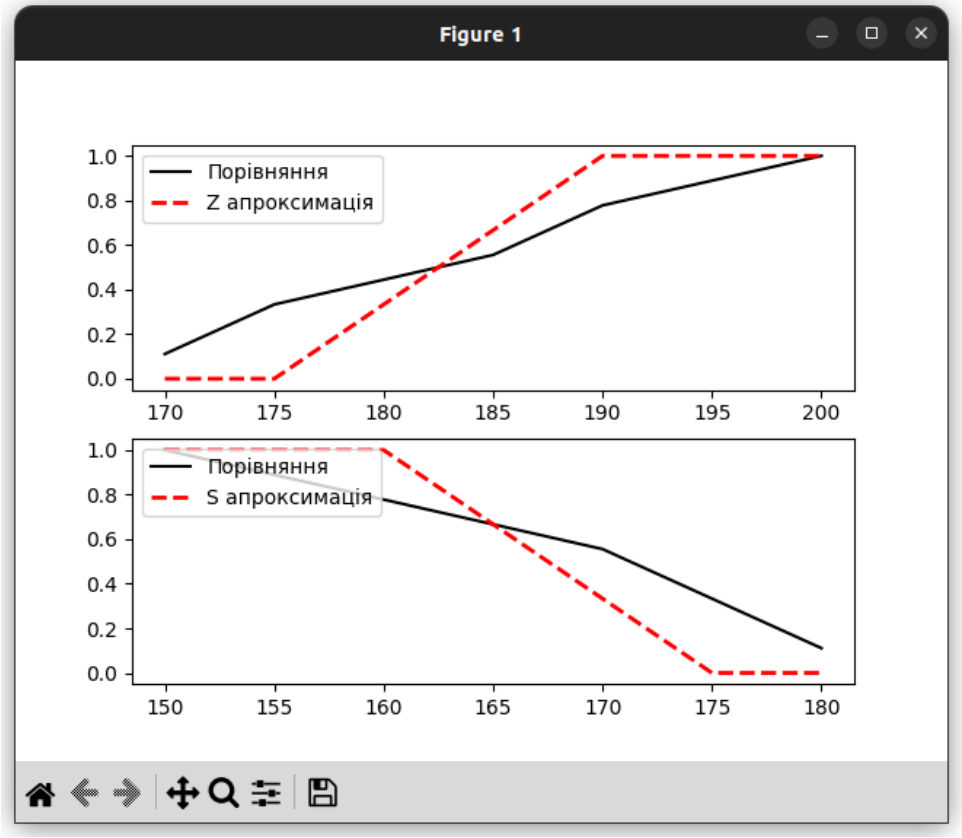
2. Визначити ступінь приналежності і-го елемента до нечіткої множини, який відповідає і-й координаті власного вектора матриці парних порівнянь.

3. Створити програму, яка розраховує ступінь приналежності на основі матриці попарних порівнянь, нормалізує функцію приналежності, будує її графік і апроксимує однією з S-подібних або Z-подібних функцій.

4. Визначити носій нечіткої множини, її ядро та границі.

Виконання завдання:

```
Вектори
[0.027027027027027032, 0.08108108108108109, 0.10810810810810811, 0.13513513513513514, 0.1891891891891892, 0.21621621621621623, 0.24324324324324328]
[0.23076923076923078, 0.20512820512820515, 0.1794871794871795, 0.15384615384615388, 0.12820512820512822, 0.07692307692307694, 0.025641025641025647]
Вектори після нормалалізації
[0.11111111111111112, 0.3333333333333333, 0.4444444444444444, 0.5555555555555555, 0.7777777777777777, 0.8888888888888888, 1.0]
[1.0, 0.888888888888889, 0.7777777777777778, 0.6666666666666667, 0.5555555555555556, 0.33333333333333337, 0.11111111111111113]
```



```
Для матриці 1
Ядро
[200]
Межі
[170, 175, 180, 185, 190, 195]
Носій
[170, 175, 180, 185, 190, 195, 200]
Для матриці 2
Ядро
[150]
Межі
[155, 160, 165, 170, 175, 180]
Носій
[150, 155, 160, 165, 170, 175, 180]
```

Висновок:

Виконавши лабораторну роботу я навчився будувати непрямим методом побудови функцій приналежності нечіткої множини на основі попарних порівнянь елементів нечіткої множини експертом.

Додаток:

```
from matplotlib import pyplot as plt


def aprocZ(x: int, a: int, b: int) -> float:
    if x <= a:
        return 1
    elif a < x < b:
        return (b - x) / (b - a)
    else:
        return 0


def aprocS(x: int, a: int, b: int) -> float:
    if x <= a:
        return 0
    elif a < x < b:
        return (x - a) / (b - a)
    else:
        return 1


def carrier(array_label: list[int], array_mu: list[float]):
    temp: list[int] = []
    for i in range(0, len(array_mu)):
        if array_mu[i]:
            temp.append(array_label[i])
    print("Носій")
    print(temp)


def core(array_label: list[int], array_mu: list[float]):
    temp: list[int] = []
    for i in range(0, len(array_mu)):
```

```
        if array_mu[i] == 1:
            temp.append(array_label[i])
    print("Ядро")
    print(temp)


def border(array_label: list[int], array_mu: list[float]):
    temp: list[int] = []
    for i in range(0, len(array_mu)):
        if 0 < array_mu[i] < 1.0:
            temp.append(array_label[i])
    print("Межі")
    print(temp)


def findvec(matrix: list[list[float]]) -> list[float]:
    temp: list[float] = []
    answer: list[float] = []
    size: int = len(matrix)
    temp_sum: float = 0
    for i in range(0, size):
        temp1: float = 1
        for x in matrix[i]:
            temp1 *= x
        temp.append(temp1 ** (1/size))
        temp_sum += temp[-1]
    for x in temp:
        answer.append(x/temp_sum)
    return answer


def normalvec(_vec: list[float]) -> list[float]:
    maxelem = max(_vec)
    answer: list[float] = [x/maxelem for x in _vec]
    return answer


matrix1: list[list[float]] = []
matrix2: list[list[float]] = []
label_array1: list[int] = [x for x in range(170, 205, 5)]
label_array2: list[int] = [x for x in range(150, 185, 5)]
weight_array1: list[int] = [1, 3, 4, 5, 7, 8, 9]
weight_array2: list[int] = [9, 8, 7, 6, 5, 3, 1]
for i in weight_array1:
    matrix1.append([i/x for x in weight_array1])
for i in weight_array2:
    matrix2.append([i/x for x in weight_array2])
array_mu1: list[float] = findvec(matrix1)
array_mu2: list[float] = findvec(matrix2)
print("Вектори")
print(array_mu1)
print(array_mu2)
print("Вектори після нормалалізації")
array_mu1 = normalvec(array_mu1)
array_mu2 = normalvec(array_mu2)
print(array_mu1)
print(array_mu2)
print("Для матриці 1")
core(label_array1, array_mu1)
border(label_array1, array_mu1)
carrier(label_array1, array_mu1)
print("Для матриці 2")
core(label_array2, array_mu2)
```

```
border(label_array2, array_mu2)
carrier(label_array2, array_mu2)
array_aprox1 = [aprocS(x, 175, 190) for x in label_array1]
array_aprox2 = [aprocZ(x, 160, 175) for x in label_array2]
fig, ax = plt.subplots(2)
ax[0].plot(label_array1, array_mu1, color='#000000', label='Порівняння')
ax[0].plot(label_array1, array_aprox1, linestyle='--', color='#FF0000', linewidth=2, label='Z апроксимація')
ax[0].legend(loc=2)
ax[1].plot(label_array2, array_mu2, color='#000000', label='Порівняння')
ax[1].plot(label_array2, array_aprox2, linestyle='--', color='#FF0000', linewidth=2, label='S апроксимація')
ax[1].legend(loc=2)
plt.show()
```