**Submission of 2nd Assignment**

***“Defuzzification Methods”***

**Dhruv Roy Talukdar**

**Soft Computing**

1) Max membership principle

n = int(input("Enter the number of fuzzy sets:\n"))

mp = {}

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    for i in range(0, len(v)):

        if v[i] in mp.keys():

            mp[v[i]] = max(mp[v[i]], d[i])

        else:

            mp[v[i]] = d[i]

values = list(mp.keys())

degrees = list(mp.values())

res = []

for i in range(0, len(values)):

    if degrees[i] == max(degrees):

        res.append(values[i])

print("The defuzzified value(s) are", res)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value(s) are [6.0, 7.0]

2) Centroid Method

n = int(input("Enter the number of fuzzy sets:\n"))

mp = {}

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    for i in range(0, len(v)):

        if v[i] in mp.keys():

            mp[v[i]] = max(mp[v[i]], d[i])

        else:

            mp[v[i]] = d[i]

values = list(mp.keys())

degrees = list(mp.values())

INTERVAL = 0.08

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

areaXxbar = 0.0

area = 0.0

for i in range(0, len(values)-1):

    x1 = values[i]

    while x1 < values[i+1]:

        x2 = x1 + INTERVAL

        curr\_area = (

            INTERVAL\*(y\_value(x1, values[i+1], degrees[i+1], values[i], degrees[i]) + y\_value(x2, values[i+1], degrees[i+1], values[i], degrees[i]))/2)

        area += curr\_area

        areaXxbar += (curr\_area \* ((x1+x2)/2))

        x1 += INTERVAL

print("The defuzzified value is ", areaXxbar/area)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value is  4.948717948717948

3) Mean max membership

n = int(input("Enter the number of fuzzy sets:\n"))

mp = {}

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    for i in range(0, len(v)):

        if v[i] in mp.keys():

            mp[v[i]] = max(mp[v[i]], d[i])

        else:

            mp[v[i]] = d[i]

values = list(mp.keys())

degrees = list(mp.values())

INTERVAL = 0.08

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

li = []

for i in range(0, len(values)-1):

    if degrees[i] == max(degrees):

        li.append(values[i])

print("The defuzzified value is ", (max(li)+min(li))/2)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value is  6.5

4) Center of sums

n = int(input("Enter the number of fuzzy sets:\n"))

values = []

degrees = []

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    values.append(v)

    degrees.append(d)

INTERVAL = 0.08

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

def find\_area(*value1*, *degree1*, *value2*, *degree2*):

    area = 0.0

    x1 = value1

    while x1 < value2:

        x2 = x1 + INTERVAL

        curr\_area = (

            INTERVAL\*(y\_value(x1, value2, degree2, value1, degree1) + y\_value(x2, value2, degree2, value1, degree1))/2)

        area += curr\_area

        x1 += INTERVAL

    return area

ans = 0.0

denom = 0.0

for i in range(0, n):

    v\_list = values[i]

    d\_list = degrees[i]

    area = 0.0

    for j in range(0, len(v\_list)-1):

        area += find\_area(v\_list[j], d\_list[j], v\_list[j+1], d\_list[j+1])

    ans += (area \* (sum(v\_list)/len(v\_list)))

    denom += (area)

ans /= denom

print("The defuzzified value is ", ans)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value is  5.0

5) First (or Last) Of Maxima Method

n = int(input("Enter the number of fuzzy sets:\n"))

mp = {}

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    for i in range(0, len(v)):

        if v[i] in mp.keys():

            mp[v[i]] = max(mp[v[i]], d[i])

        else:

            mp[v[i]] = d[i]

values = list(mp.keys())

degrees = list(mp.values())

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

li = []

for i in range(0, len(values)-1):

    if degrees[i] == max(degrees):

        li.append(values[i])

print(f"First of maxima {min(li)} and last of maxima {max(li)}")

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# First of maxima 6.0 and last of maxima 7.0

6) Weighted Average Method

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

n = int(input("Enter the number of fuzzy sets:\n"))

mul = 0.0

degrees = 0.0

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    avg = sum(v)/len(v)

    for i in range(0, len(v)-1):

        if v[i] <= avg <= v[i+1]:

            y\_val = y\_value(avg, v[i+1], d[i+1], v[i], d[i])

    mul += (avg \* y\_val)

    degrees += y\_val

print("The defuzzified value is ", mul/degrees)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value is  5.416666666666667

7) Center of Largest Area

n = int(input("Enter the number of fuzzy sets:\n"))

values = []

degrees = []

for i in range(0, n):

    v = [float(i) for i in input(

        "Enter the values of set space separated:\n").split(" ")]

    d = [float(i)

         for i in input("Enter the degrees space separated:\n").split(" ")]

    assert(len(v) == len(d))

    values.append(v)

    degrees.append(d)

INTERVAL = 0.08

def y\_value(*x\_val*, *x2*, *y2*, *x1*, *y1*):

    slope = float(y2-y1)/(x2-x1)

    b = y2 - x2\*slope

    return (x\_val\*slope + b)

def find\_area(*value1*, *degree1*, *value2*, *degree2*):

    area = 0.0

    x1 = value1

    while x1 < value2:

        x2 = x1 + INTERVAL

        curr\_area = (

            INTERVAL\*(y\_value(x1, value2, degree2, value1, degree1) + y\_value(x2, value2, degree2, value1, degree1))/2)

        area += curr\_area

        x1 += INTERVAL

    return area

index = 0

store = -1

for i in range(0, n):

    v\_list = values[i]

    d\_list = degrees[i]

    area = 0.0

    for j in range(0, len(v\_list)-1):

        area += find\_area(v\_list[j], d\_list[j], v\_list[j+1], d\_list[j+1])

    if store < area:

        store = area

        index = i

val = sum((values[index])) / len(values[index])

print("The defuzzified value is ", val)

# OUTPUT

# Enter the number of fuzzy sets:

# 3

# Enter the values of set space separated:

# 0 1 2 3 4 5

# Enter the degrees space separated:

# 0 0.3 0.3 0.3 0.3 0

# Enter the values of set space separated:

# 3 4 5 6 7

# Enter the degrees space separated:

# 0 0.5 0.5 0.5 0

# Enter the values of set space separated:

# 5 6 7 8

# Enter the degrees space separated:

# 0 1 1 0

# The defuzzified value is  6.5