FUZZY LOGIC UNTUK MEMERINGKATKAN RESTORAN BERDASARKAN NILAI MAKANAN DAN PELAYANA

1. Identifikasi Masalah

Perkembangan teknologi akhir ini semakin merata di berbagai aspek, Artificial Intelligence (AI) diklasifikasikan sebagai kecerdasan yang dibuat oleh mesin. Kecerdasan buatan dihasilkan dan diprogram ke dalam suatu sistem (komputer) untuk melakukan tugas-tugas yang dapat dilakukan manusia. Prinsip kebenaran parsial diperkenalkan dalam fuzzy logic, yang meningkatkan boolean logic. Tingkat kebenaran menggantikan kebenaran boolean dalam fuzzy logic. Karena ilmu fuzzy logic modern dan metodis yang baru ditemukan beberapa tahun yang lalu, fuzzy logic dikatakan sebagai logika baru yang lama, meskipun pada kenyataannya ide fuzzy logic telah ada sejak lama.

Tujuan dari penulisan laporan ini adalah untuk memahami Fuzzy Logic serta penerapannya, memahami nama linguistik Fuzzy Logic, memahami fungsi keanggotaan Fuzzy Logic, memahami tahapan-tahapan yang ada pada Fuzzy Logic.

Diberikan sampel data berupa excel kemudian diminta untuk menentukan 10 restoran terbaik berdasarkan nilai pelayanan dan nilai makanan. Nilai pelayanan memiliki semesta [0-100] dan nilai makanan dengan semesta [0-10]. Penentuan 10 terbaik ini dilakukan menggunakan metode fuzzy logic.

Sampel data

id	pelayanan	makanan
1	58	7
2	54	1
3	98	2
4	52	4
5	11	4
6	59	10
7	61	8
8	30	10
9	45	1
10	36	9
11	10	5
12	38	7
13	80	3
14	31	8
15	78	5
16	82	6
17	70	3
18	3	9
19	42	3
20	49	10

2. Pembahasan

2.1. Pengertian Fuzzy Logic

Fuzzy Logic adalah suatu logika yang mempunyai nilai kesamaan yang membuat sebuah komputer dapat meniru kecerdasan manusia dengan harapan sesuatu yang dikerjakan manusia dapat dikerjakan oleh komputer. Fuzzy Logic yang didasari konsep himpunan fuzzy berarti memetakan suatu ruang input ke dalam suatu ruang output. Fuzzy Logic memiliki implementasinya yang luas baik di bidang engineering, psikologi, sosial, dan juga bidang ekonomi misal robot, peralatan rumah tangga, kendaraan, dan lain-lain.

2.2. Jumlah dan Nama Linguistik

Dalam menentukan variabel linguistik itu tergantung pada tingkat perbedaan. Jika kita menggunakan lebih banyak variabel, maka kita memiliki akurasi yang lebih tinggi. Nama linguistik untuk input pada fuzzy logic ini ada **3** untuk variabel fuzzy **pelayanan** diantaranya **buruk, biasa** dan **istimewa**. Sementaran untuk variable fuzzy **makanan** terdapat **4** nama linguistik yaitu, **tengik, layak, baik,** dan **enak**. Sedangkan untuk output fuzzy ini, nama variabel fuzzynya adalah **peringkat** dengan 2 nama linguistik, yaitu **jelek** dan **terbaik.**

Variabel	Variable fuzzy	Domain	Keterangan
Input	Pelayanan	0-100	Nilai

	Makanan	0-10	Nilai
Output	Peringkat	0-100	%

2.3. Bentuk dan Batas Fungsi Keanggotaan Input

Fungsi keanggotaan dibuat dalam representasi linear, yaitu permukaan digambarkan sebagai suatu garis lurus. Bentuk ini paling sederhana dan menjadi pilihan yang baik untuk mendekati suatu konsep yang kurang jelas. Bentuk yang digunakan pada fuzzy ini, yaitu trapesium dan segitiga. Ada 2 kemungkinan keadaan himpunan fuzzy yang linier. Pertama, kenaikan himpunan dimulai pada nilai dominan yang memiliki derajat keanggotaan nol [0] bergerak kekanan menuju ke nilai domain yang memiliki derajat keanggotaan lebih tinggi.

Pengendali Fuzzy yang dirancang menggunakan dua masukan yaitu pelayanan dan makanan serta satu keluaran yaitu peringkat. Berdasarkan data pada "restoran.xlsx" dapat ditentukan range untuk setiap nama linguistik pelayanan dan makanan sebagai berikut:

Variabel	Variabel fuzzy	Nama linguistik	Batasan	Bentuk
Input	Pelayanan	Buruk	[0, 0, 20, 40]	Trapesium
		Biasa	[20, 40, 80]	Segitiga
		Istimewa	[60, 80, 100, 100]	Trapesium
	Makanan	Tengik	[0, 0, 2, 3]	Trapesium
		Layak	[2, 3, 5]	Segitiga
		Baik	[4, 5, 7]	Segitiga
		Enak	[6, 9, 10, 10]	Trapesium
Output	Peringkat	Jelek	[0, 0, 50, 80]	Trapesium

		Terbaik	[50, 80, 100, 100]	Trapesium

Fungsi segitiga

$$\mu(x) = \begin{cases} 0; x \leq a \ atau \ x \geq c \\ \dfrac{x-a}{b-a}; \quad a \leq x \leq b \\ \dfrac{-(x-c)}{c-b}; \quad b \leq x \leq c \end{cases}$$

Fungsi trapesium

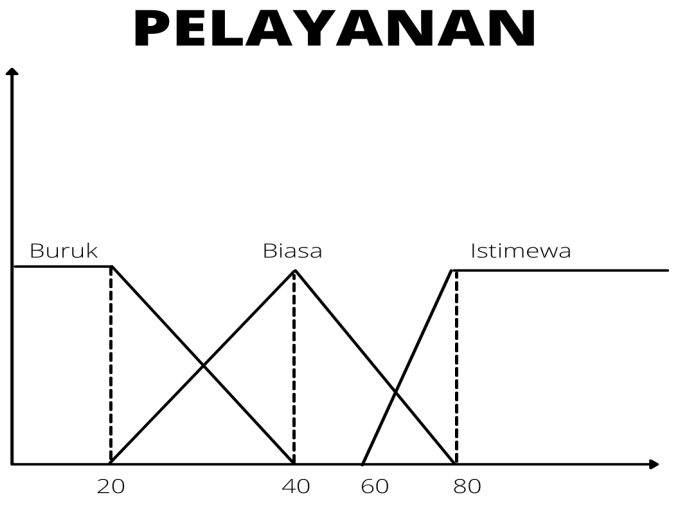
$$\mu(x) = \begin{cases} 0; x \leq a \ atau \quad x \geq d \\ \dfrac{x-a}{b-a}; a \leq x \leq b \\ 1; b \leq x \leq c \\ \dfrac{d-x}{d-c}; c \leq x \leq d \end{cases}$$

Source code bentuk fungsi keanggotaan

```
def trapesium(x,a,b,c,d):
    if ((x<=a) or (x>=d)):
      nilai = 0
    elif ((x>a) and (x<b)):
      nilai = (x-a)/(b-a)
    elif ((x>=b) and (x<=c)):
      nilai = 1
    elif ((x>c) and (x<=d)):
      nilai = -(x-d)/(d-c)
    return nilai
```

```
def segitiga(x,a,b,c):
    if ((x<=a) or (x>=c)):
        nilai = 0
    elif ((x>a) and (x<=b)):
        nilai = (x-a)/(b-a)
    elif ((x>b) and (x<=c)):
        nilai = -(x-c)/(c-b)
    return nilai
```

- **Fungsi Keanggotaan Pelayanan**



Pada variabel fuzzy pelayanan terdapat 2 bentuk fungsi keanggotaan, yaitu trapesium dan segitiga. Keanggotaan fungsi trapesium

ditentukan oleh 4 parameter [a, b, c, d]. Sementara fungsi keanggotaan segitiga ditentukan oleh 3 parameter [a, b, c]. Fungsi keanggotaan buruk dan istimewa dibuat dengan bentuk trapesium dan fungsi keanggotaan biasa dibuat dengan bentuk segitiga. Fungsi keanggotaan trapesium dan segitiga digunakan karena sederhana dan penggunaan fungsi kompleks tidak menambah presisi lebih dalam keluaran.

Setiap himpunan fuzzy yang digunakan mendeskripsikan parameter pelayanan. Fuzzy set buruk dibuat lebih sempit, yaitu [0, 0, 20, 40] daripada fuzzy set yang lain. Tentulah sangat tidak nyaman sekali pergi ke restoran yang memiliki pelayanan yang buruk. Cara mendesain nilai-nilai parameter seperti itu bergantung pada intuisi dan pengalaman perancang itu sendiri. Fungsi keanggotaan ini akan menghasilkan derajat keanggotaan yang menggambarkan tingkat keanggotaan pada himpunan fuzzy.

Source code

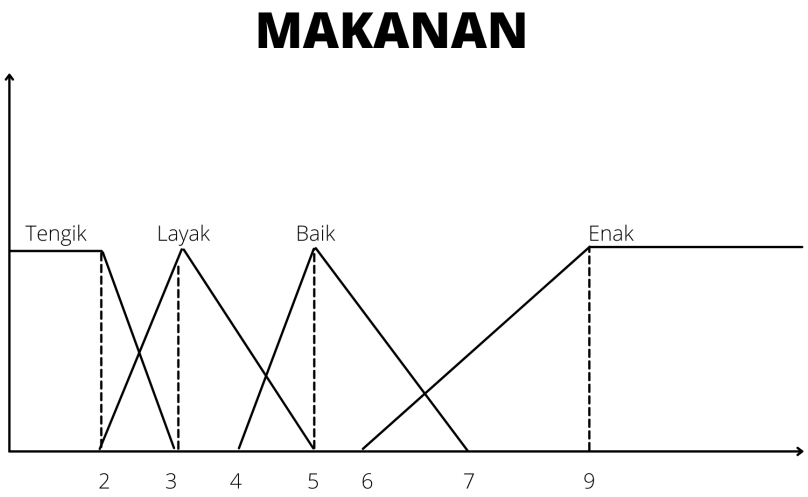
```
def buruk(x,derajatKeanggotaan):
    a = 0
    b = 0
    c = 20
    d = 40
    derajatKeanggotaan['buruk'] = round(trapesium(x,a,b,c,d),2)
    return derajatKeanggotaan
```

```
def biasa(x,derajatKeanggotaan):
    a = 20
    b = 40
    c = 80
    derajatKeanggotaan['biasa'] = round(segitiga(x,a,b,c),2)
    return derajatKeanggotaan
```

```
def istimewa(x,derajatKeanggotaan):
    a = 70
    b = 80
    c = 100.1
    d = 100.1
    derajatKeanggotaan['istimewa'] = round(trapesium(x,a,b,c,d),2)
    return derajatKeanggotaan
```

```
{1: {'biasa': 0.55, 'buruk': 0, 'istimewa': 0},
 2: {'biasa': 0.65, 'buruk': 0, 'istimewa': 0},
 3: {'biasa': 0, 'buruk': 0, 'istimewa': 1},
 4: {'biasa': 0.7, 'buruk': 0, 'istimewa': 0},
 5: {'biasa': 0, 'buruk': 1, 'istimewa': 0},
 6: {'biasa': 0.53, 'buruk': 0, 'istimewa': 0},
 7: {'biasa': 0.47, 'buruk': 0, 'istimewa': 0},
 8: {'biasa': 0.5, 'buruk': 0.5, 'istimewa': 0},
 9: {'biasa': 0.88, 'buruk': 0, 'istimewa': 0},
 10: {'biasa': 0.8, 'buruk': 0.2, 'istimewa': 0},
 11: {'biasa': 0, 'buruk': 1, 'istimewa': 0},
 12: {'biasa': 0.9, 'buruk': 0.1, 'istimewa': 0},
 13: {'biasa': 0, 'buruk': 0, 'istimewa': 1},
 14: {'biasa': 0.55, 'buruk': 0.45, 'istimewa': 0},
 15: {'biasa': 0.05, 'buruk': 0, 'istimewa': 0.8},
 16: {'biasa': 0, 'buruk': 0, 'istimewa': 1},
 17: {'biasa': 0.25, 'buruk': 0, 'istimewa': 0},
 18: {'biasa': 0, 'buruk': 1, 'istimewa': 0},
 19: {'biasa': 0.95, 'buruk': 0, 'istimewa': 0},
 20: {'biasa': 0.78, 'buruk': 0, 'istimewa': 0},
```

- **Fungsi Keanggotaan Makanan**



Pada parameter fuzzy makanan digunakan fungsi keanggotaan segitiga ditentukan oleh 3 parameter [a, b, c] dan juga fungsi keanggotaan trapesium ditentukan oleh 4 parameter [a, b, c, d]. Dalam fungsi keanggotaan makanan terdapat 4 nama linguistik yaitu tengik dengan bentuk trapesium batasnya [0, 0, 2, 3]. Layak memiliki bentuk segitiga dengan batas [2, 3, 5]. Baik bentuknya segitiga batasnya [4, 5, 7]. Kemudian enak bentuknya trapesium dengan batas [6, 9, 10, 10].

Source code

```
def tengik(x,derajatKeanggotaan):
    a = 0
    b = 0
    c = 2
    d = 3
    derajatKeanggotaan['tengik'] = round(trapesium(x,a,b,c,d),2)
    return derajatKeanggotaan
```

```
def layak(x,derajatKeanggotaan):
    a = 2
    b = 3
    c = 5
    derajatKeanggotaan['layak'] = round(segitiga(x,a,b,c),2)
    return derajatKeanggotaan
```

```
def baik(x,derajatKeanggotaan):
    a = 4
    b = 5
    c = 7
    derajatKeanggotaan['baik'] = round(segitiga(x,a,b,c),2)
    return derajatKeanggotaan
```

```
def enak(x,derajatKeanggotaan):
    a = 6
    b = 9
    c = 10.1
    d = 10.1
    derajatKeanggotaan['enak'] = round(trapesium(x,a,b,c,d),2)
    return derajatKeanggotaan
```

```
{1: {'baik': 0, 'enak': 0.33, 'layak': 0, 'tengik': 0},
 2: {'baik': 0, 'enak': 0, 'layak': 0, 'tengik': 1},
 3: {'baik': 0, 'enak': 0, 'layak': 0, 'tengik': 1},
 4: {'baik': 0, 'enak': 0, 'layak': 0.5, 'tengik': 0},
 5: {'baik': 0, 'enak': 0, 'layak': 0.5, 'tengik': 0},
 6: {'baik': 0, 'enak': 1, 'layak': 0, 'tengik': 0},
 7: {'baik': 0, 'enak': 0.67, 'layak': 0, 'tengik': 0},
 8: {'baik': 0, 'enak': 1, 'layak': 0, 'tengik': 0},
 9: {'baik': 0, 'enak': 0, 'layak': 0, 'tengik': 1},
 10: {'baik': 0, 'enak': 1, 'layak': 0, 'tengik': 0},
 11: {'baik': 1.0, 'enak': 0, 'layak': 0, 'tengik': 0},
 12: {'baik': 0, 'enak': 0.33, 'layak': 0, 'tengik': 0},
 13: {'baik': 0, 'enak': 0, 'layak': 1.0, 'tengik': 0},
 14: {'baik': 0, 'enak': 0.67, 'layak': 0, 'tengik': 0},
 15: {'baik': 1.0, 'enak': 0, 'layak': 0, 'tengik': 0},
 16: {'baik': 0.5, 'enak': 0, 'layak': 0, 'tengik': 0},
 17: {'baik': 0, 'enak': 0, 'layak': 1.0, 'tengik': 0},
 18: {'baik': 0, 'enak': 1, 'layak': 0, 'tengik': 0},
 19: {'baik': 0, 'enak': 0, 'layak': 1.0, 'tengik': 0},
 20: {'baik': 0, 'enak': 1, 'layak': 0, 'tengik': 0},
```

2.4. Aturan Inferensi

Aturan inferensi dibuat untuk mengontrol output. Aturan ini akan mengikuti logika dan pengetahuan heuristik. Berikut aturan inferensi yang digunakan dalam fuzzy ini sebagai berikut:

- Jika pelayanan BURUK dan kualitas makanan TENGIK maka peringkat JELEK
- Jika pelayanan BURUK dan kualitas makanan LAYAK maka peringkat JELEK
- Jika pelayanan BURUK dan kualitas makanan BAIK maka peringkat JELEK
- Jika pelayanan BURUK dan kualitas makanan ENAK maka peringkat JELEK
- Jika pelayanan BIASA dan kualitas makanan TENGIK maka peringkat JELEK
- Jika pelayanan BIASA dan kualitas makanan LAYAK maka peringkat TERBAIK
- Jika pelayanan BIASA dan kualitas makanan BAIK maka peringkat TERBAIK
- Jika pelayanan BIASA dan kualitas makanan ENAK maka peringkat TERBAIK
- Jika pelayanan ISTIMEWA dan kualitas makanan TENGIK maka peringkat JELEK
- Jika pelayanan ISTIMEWA dan kualitas makanan LAYAK maka peringkat TERBAIK
- Jika pelayanan ISTIMEWA dan kualitas makanan BAIK maka peringkat TERBAIK
- Jika pelayanan ISTIMEWA dan kualitas makanan ENAK maka peringkat TERBAIK

Pelayanan\Makanan	Tengik	Layak	Baik	Enak
Buruk	Jelek	Jelek	Jelek	Jelek
Biasa	Jelek	Terbaik	Terbaik	Terbaik
Istimewa	Jelek	Terbaik	Terbaik	Terbaik

Dengan metode inferensi Mamdani, diperoleh proses inferensi dengan menggunakan aturan conjunction (^) terhadap kedelapan aturan baru di atas, untuk mengambil derajat keanggotaan minimum dari nilai linguistik yang ada. Setelah itu gunakan aturan disjunction (V) untuk menentukan nilai derajat keanggotaan maksimum dari nilai-nilai linguistik.

Source code

```python
def inference(nilai_pelayanan,nilai_makanan):
        nilaiStatus = {}
        i = 1
        while i <=100:
            status = {}
            terbaik = []
            jelek = []
            if ((nilai_pelayanan[i]['buruk']!=0) and (nilai_makanan[i]['tengik']!=0)):
                jelek.append(min(nilai_pelayanan[i]['buruk'],nilai_makanan[i]['tengik']))
            if ((nilai_pelayanan[i]['buruk']!=0) and (nilai_makanan[i]['layak']!=0)):
                jelek.append(min(nilai_pelayanan[i]['buruk'],nilai_makanan[i]['layak']))
            if ((nilai_pelayanan[i]['buruk']!=0) and (nilai_makanan[i]['baik']!=0)):
                jelek.append(min(nilai_pelayanan[i]['buruk'],nilai_makanan[i]['baik']))
            if ((nilai_pelayanan[i]['buruk']!=0) and (nilai_makanan[i]['enak']!=0)):
                jelek.append(min(nilai_pelayanan[i]['buruk'],nilai_makanan[i]['enak']))
            if ((nilai_pelayanan[i]['biasa']!=0) and (nilai_makanan[i]['tengik']!=0)):
                jelek.append(min(nilai_pelayanan[i]['biasa'],nilai_makanan[i]['tengik']))
            if ((nilai_pelayanan[i]['biasa']!=0) and (nilai_makanan[i]['layak']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['biasa'],nilai_makanan[i]['layak']))
            if ((nilai_pelayanan[i]['biasa']!=0) and (nilai_makanan[i]['baik']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['biasa'],nilai_makanan[i]['baik']))
            if ((nilai_pelayanan[i]['biasa']!=0) and (nilai_makanan[i]['enak']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['biasa'],nilai_makanan[i]['enak']))
            if ((nilai_pelayanan[i]['istimewa']!=0) and (nilai_makanan[i]['tengik']!=0)):
                jelek.append(min(nilai_pelayanan[i]['istimewa'],nilai_makanan[i]['tengik']))
            if ((nilai_pelayanan[i]['istimewa']!=0) and (nilai_makanan[i]['layak']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['istimewa'],nilai_makanan[i]['layak']))
            if ((nilai_pelayanan[i]['istimewa']!=0) and (nilai_makanan[i]['baik']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['istimewa'],nilai_makanan[i]['baik']))
            if ((nilai_pelayanan[i]['istimewa']!=0) and (nilai_makanan[i]['enak']!=0)):
                terbaik.append(min(nilai_pelayanan[i]['istimewa'],nilai_makanan[i]['enak']))
            status['terbaik'] = terbaik
            status['jelek'] = jelek
            nilaiStatus[i] = status
            i+=1
        for i in nilaiStatus:
            if (nilaiStatus[i]['jelek']==[]):
                nilaiStatus[i]['jelek'] = [0]
            elif(nilaiStatus[i]['terbaik']==[]):
                nilaiStatus[i]['terbaik'] = [0]
            nilaiStatus[i]['terbaik'] = max(nilaiStatus[i]['terbaik'])
            nilaiStatus[i]['jelek'] = max(nilaiStatus[i]['jelek'])
        return nilaiStatus
```

```
{1: {'jelek': 0, 'terbaik': 0.33},
 2: {'jelek': 0.65, 'terbaik': 0},
 3: {'jelek': 1, 'terbaik': 0},
 4: {'jelek': 0, 'terbaik': 0.5},
 5: {'jelek': 0.5, 'terbaik': 0},
 6: {'jelek': 0, 'terbaik': 0.53},
 7: {'jelek': 0, 'terbaik': 0.47},
 8: {'jelek': 0.5, 'terbaik': 0.5},
 9: {'jelek': 0.88, 'terbaik': 0},
 10: {'jelek': 0.2, 'terbaik': 0.8},
 11: {'jelek': 1, 'terbaik': 0},
 12: {'jelek': 0.1, 'terbaik': 0.33},
 13: {'jelek': 0, 'terbaik': 1},
 14: {'jelek': 0.45, 'terbaik': 0.55},
 15: {'jelek': 0, 'terbaik': 0.8},
 16: {'jelek': 0, 'terbaik': 0.5},
 17: {'jelek': 0, 'terbaik': 0.25},
 18: {'jelek': 1, 'terbaik': 0},
 19: {'jelek': 0, 'terbaik': 0.95},
 20: {'jelek': 0, 'terbaik': 0.78},
```

2.5. Metode Defuzzification

Defuzzifikasi adalah proses mendapatkan nilai crisp dari suatu himpunan fuzzy. Proses inferensi fuzzy tipe Mamdani terdiri dari lima langkah:

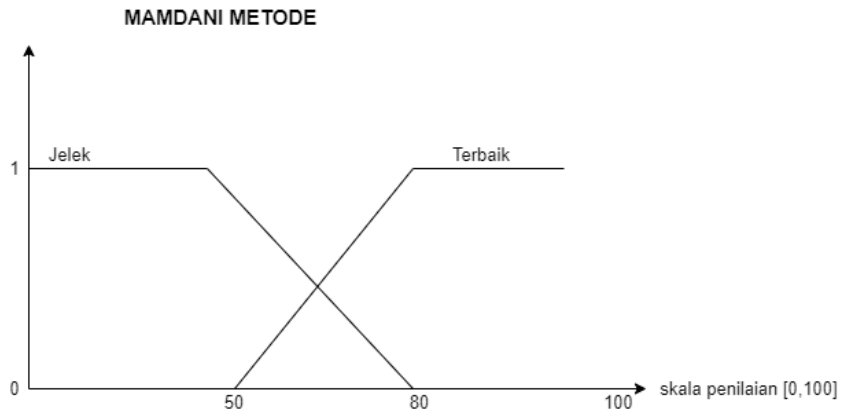
Langkah 1: Fuzzifikasi variabel input
Langkah 2: Terapkan operator fuzzy
Langkah 3: Terapkan metode implikasi
Langkah 4: Terapkan metode agregasi
Langkah 5: Defuzzifikasi

Di antara banyak metode defuzzifikasi yang ada. Metode Centroid (juga disebut pusat area atau pusat gravitasi) yang merupakan defuzzifikasi yang paling umum dan paling menarik secara fisik metode (Sugeno, 1985; Lee, 1990). Pada Metode Mamdani, untuk mendapatkan nilai tersebut digunakan Metode Centroid atau mencari bobot nilai tengah kurva daerah fuzzy (center of gravity). Metode yang digunakan yaitu, metode mamdani/centroid karena metode ini merupakan metode yang paling rinci, akurat, dan manusiawi dalam pengambilan keputusannya. Dengan demikian, cocok untuk memutuskan 10 restoran terbaik berdasarkan pelayanan dan kualitas makanan. Metode ini menghitung nilai crisp menggunakan rumus berikut:

$$y^* = \frac{\int y \mu_R(y) dy}{\int \mu_R(y) dy}$$

2.6. Bentuk dan Batas Fungsi Keanggotaan Output

Fungsi keanggotaan output dibuat dalam bentuk trapesium dengan batas [0, 0, 50, 80] untuk variabel linguistik jelek dan untuk variabel linguistik terbaik batasnya [50, 80, 100, 100].




```python
# Metode mamdani
def defuzzification(nilaiStatus):
    hasil = {}
    for i in nilaiStatus:
        if ((nilaiStatus[i]['terbaik'] !=0) and (nilaiStatus[i]['jelek'] == 0)):
            temp = (60*(1/3)) + (65*(1/2)) +(70+80)*nilaiStatus[i]['terbaik']
            pem = (((1)/(3))+((1)/(2))+(nilaiStatus[i]['terbaik']*2))
            hasil[i] = round(temp/pem,2)
        elif ((nilaiStatus[i]['jelek'] !=0) and (nilaiStatus[i]['terbaik'] == 0)):
            temp = ((10+20+30+40+50)*nilaiStatus[i]['jelek'])+(60*(-((-2))/(3)))+(65*(-(-2)/(3)))+(70*(-(-1)/(3)))
            pem = (-(-2)/(3))+(-(-2)/(3))+(nilaiStatus[i]['jelek']*5)
            hasil[i] = round(temp/pem,2)
        else:
            temp = (((10+20+30+40+50+60)*nilaiStatus[i]['jelek'])) + (((70+80+90+100)*nilaiStatus[i]['terbaik']))
            pem = (6*(nilaiStatus[i]['jelek'])) + (4*(nilaiStatus[i]['terbaik']))
            hasil[i] = round(temp/pem,2)
    return hasil
```

```python
[ ] if __name__=="__main__":
    data_restoran = pd.read_excel('./restoran.xlsx')
    pelayanan = pd.DataFrame(data_restoran, columns=['pelayanan'])
    makanan = pd.DataFrame(data_restoran, columns=['makanan'])
    nilai_makanan, nilai_pelayanan = fuzzification(pelayanan,makanan)
    nilaiStatus = inference(nilai_pelayanan,nilai_makanan)
    hasil = defuzzification(nilaiStatus)
    peringkat = sorted(hasil.items(),key=lambda x: [1])
    excel = []
    for i in range(len(peringkat)):
        excel.append(peringkat[i][0])
    myexcel = []
    for i in reversed(excel[-10:]):
        myexcel.append(i)
    df = pd.DataFrame(myexcel)
    df.to_excel("peringkat.xls",index=False,header=False)
```

Output

Ini adalah 10 id restoran terbaik.

A
80
71
92
91
79
69
52
42
24
13

2.7. Simpulan

Fuzzy logic merupakan suatu cara untuk membantu pembuatan keputusan dengan menggunakan logika samar antara [0,1] yang melalui proses fuzzification, inference dan defuzzification. Dalam prosesnya nilai input diubah menjadi sebuah variabel linguistik dan tentukan batas serta bentuk fungsi keanggotaannya sehingga didapat derajat keanggotaan nilai tersebut. Setelah itu dibuat aturan inferensi yang akan mengontrol output keluaran sehingga sesuai dengan fungsi keanggotaan output yang telah ditentukan. Dalam proses fuzzy ini digunakan metode inferensi mamdani dengan mengambil MIN dari setiap nilai derajat keanggotaan inferensi kemudian mengambil MAX dari derajat keanggotaan linguistik. Hasil dari inferensi tersebut, lalu digunakan untuk defuzzification dengan mengubah nama linguistik dan derajat keanggotaan dengan metode center of gravity menjadi sebuah nilai peringkat yang digunakan untuk menentukan 10 restoran terbaik berdasarkan data.

SOURCE CODE :
https://colab.research.google.com/drive/1ZbkYmrt4g44UkX517M4mJyt_T3Ozw67b?usp=sharing

VIDEO PRESENTASI : https://youtu.be/e1ydria30lY

DAFTAR PUSTAKA

Sambariya, D. K. (2016, May 2). Selection of Membership Functions Based on Fuzzy Rules to Design an Efficient Power System Stabilizer. International Journal of Fuzzy Systems. https://link.springer.com/article/10.1007/s40815-016-0197-6?error=cookies_not_supported&code=82f53330-0f69-428c-b354-3d185180822a

Sciencedirect.com. 2021. Fuzzification - an overview | ScienceDirect Topics. [online] Available at: <https://www.sciencedirect.com/topics/engineering/fuzzification> [Accessed 30 April 2021].

Elektronika-portal.com. 2021. Fuzzy Logic – Metode Mamdani – Elektronika Portal. [online] Available at: <https://elektronika-portal.com/2018/11/25/fuzzy-logic-metode-mamdani/> [Accessed 30 April 2021].

Arumsasi21.blogspot.com. 2021. [online] Available at: <http://arumsasi21.blogspot.com/2016/09/fuzzy-logic_30.html> [Accessed 30 April 2021].

Sciencedirect.com. 2021. Linguistic Variable - an overview | ScienceDirect Topics. [online] Available at: <https://www.sciencedirect.com/topics/engineering/linguistic-variable> [Accessed 30 April 2021].