LAPORAN OBSERVASI TUGAS PARALEL 3 : SISTEM FUZZY

CLARISA HASYA YUTIKA | 1301174256 | IF 41 02

A. Deskripsi Masalah

Membangun sebuah sistem berbasis fuzzy logic untuk memilih 20 influencers terbaik dari 100 data influencers yang diberikan. Dipilih berdasarkan 2 atribut yang dimiliki, yaitu Followers Count dan Engagement Rate.

B. Hal yang diobservasi

Follower Count

Low : FC ≤ 20000 | Not Low : FC > 30000

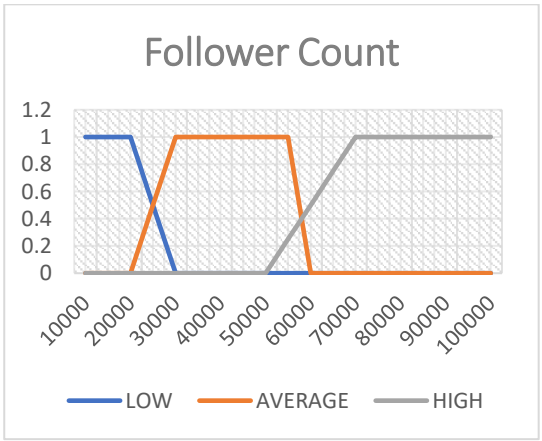
$$f(x) = \begin{cases} 1, & x \le 20000 \\ 0, & x > 30000 \\ \dfrac{30000 - x}{30000 - 20000}, & 20000 < x \le 30000 \end{cases}$$

Average : 30000 < FC ≤ 55000 | Not Average : FC ≤ 20000 or FC > 60000

$$f(x) = \begin{cases} 0, & x \le 20000, x > 60000 \\ \dfrac{x - 20000}{30000 - 20000}, & 20000 < x \le 30000 \\ 1, & 30000 < x \le 55000 \\ \dfrac{60000 - x}{60000 - 55000}, & 55000 < x \le 60000 \end{cases}$$

High : FC > 70000 | Not High : FC ≤ 50000

$$f(x) = \begin{cases} 0, & x \le 50000 \\ 1, & x > 70000 \\ \dfrac{x - 50000}{70000 - 50000}, & 50000 < x \le 70000 \end{cases}$$



Engagement Rate

Low : ER ≤ 2 | Not Low : ER > 3.5

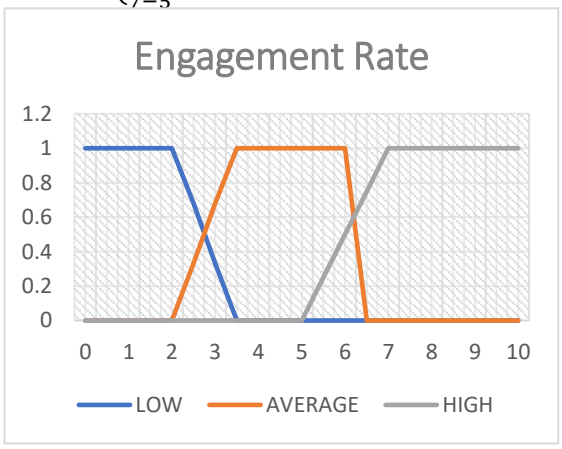
$$f(x) = \begin{cases} 1, & x \le 2 \\ 0, & x > 3.5 \\ \dfrac{3.5 - x}{3.5 - 2}, & 2 < x \le 3.5 \end{cases}$$

Average : 3.5 < ER ≤ 6 | Not Average : ER ≤ 2 or ER > 6.5

$$f(x) = \begin{cases} 0, & x \le 2, x > 6.5 \\ \dfrac{x - 2}{3.5 - 2}, & 2 < x \le 3.5 \\ 1, & 3.5 < x \le 6 \\ \dfrac{6.5 - x}{6.5 - 6}, & 6 < x \le 6.5 \end{cases}$$

High : ER > 7 | Not High : ER ≤ 5

$$f(x) = \begin{cases} 0, & x \le 5 \\ 1, & x > 7 \\ \dfrac{x-5}{7-5}, & 5 < x \le 7 \end{cases}$$



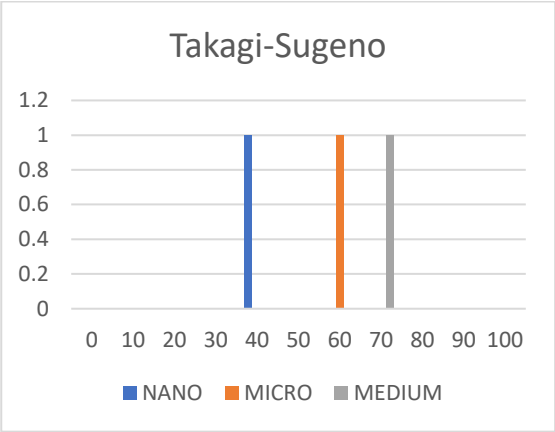
Fuzzy Rule Inferensi

Follower Count	Engagement Rate	Output
Low	Low	Nano
Low	Average	Nano
Low	High	Micro
Average	Low	Nano
Average	Average	Micro
Average	High	Medium
High	Low	Micro
High	Average	Medium
High	High	Medium

LAPORAN OBSERVASI TUGAS PARALEL 3 : SISTEM FUZZY

CLARISA HASYA YUTIKA | 1301174256 | IF 41 02

Metode Defuzzifikasi



Metode defuzzifikasi yang digunakan adalah Takagi-Sugeno dengan batas fungsi keanggotaan output untuk nano adalah 40, micro adalah 60, dan medium adalah 70

C. Proses yang dibangun

Membaca file CSV

```
def loadData():
    data = []
    with open('influencers.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count=0
        for row in csv_reader:
            if (line_count != 0):
                data.append(row)
            line_count += 1
    return data
```

Pada baris pertama terdapat isi ‘id, followerCount, engagementRate’. Supaya tidak masuk ke dalam list, sehingga pada saat line_count != 0 per baris akan diappend ke list

Fuzzifikasi

```
def followerCount(fol):
    low, avg, high = 0, 0, 0
    if (fol <= 20000):
        low = 1
    elif (fol > 30000) :
        low = 0
    elif (fol > 20000 and fol <= 30000):
        low = (30000 - fol)/(30000 - 20000)

    if (fol <= 20000 and fol > 60000):
        avg = 0
    elif (fol > 20000 and fol <= 30000):
        avg = (fol - 20000)/(30000 - 20000)
    elif (fol > 30000 and fol <= 55000):
        avg = 1
    elif (fol > 55000 and fol <= 60000):
        avg = (60000 - fol)/(60000 - 55000)

    if (fol <= 50000):
        high = 0
    elif (fol > 70000) :
        high = 1
    elif (fol > 50000 and fol <= 70000):
        high = (fol - 50000)/(70000 - 50000)

    return [low, avg, high]
```

```
def engagementRate(rate):
    low, avg, high = 0.0, 0.0, 0.0
    if (rate <= 2):
        low = 1.0
    elif (rate > 3.5) :
        low = 0.0
    elif (rate > 2 and rate <= 3.5):
        low = (3.5 - rate)/(3.5 - 2)

    if (rate <= 2  and rate > 6.5):
        avg = 0.0
    elif (rate > 2 and rate <= 3.5):
        avg = (rate - 2)/(3.5 - 2)
    elif (rate > 3.5 and rate <= 6):
        avg = 1.0
    elif (rate > 6 and rate <= 6.5):
        avg = (6.5 - rate)/(6.5 - 6)

    if (rate <= 5):
        high = 0.0
    elif (rate > 7) :
        high = 1.0
    elif (rate > 5 and rate <= 7):
        high = (rate - 5)/(7 - 5)

    return [low, avg, high]
```

Fungsi yang dibuat untuk menghitung fuzifikasi dari followerCount dan engagementRate sesuai dengan batas fungsi keanggotaan yang sudah dibuat

Inferensi

```
def inference(fol,rate):
    na, mi, me = [], [], []

    na.append(max(fol[0],rate[0])) #low, low
    na.append(max(fol[0],rate[1])) #low, average
    na.append(max(fol[1],rate[0])) #average, low

    mi.append(max(fol[0],rate[2])) #low, high
    mi.append(max(fol[1],rate[1])) #average, average
    mi.append(max(fol[2],rate[0])) #high, low

    me.append(max(fol[1],rate[2])) #average, high
    me.append(max(fol[2],rate[1])) #high, average
    me.append(max(fol[2],rate[2])) #high, high

    nano = max(na[0],na[1],na[2])
    micro = max(mi[0],mi[1],mi[2])
    medium = max(me[0],me[1],me[2])

    return [nano, micro, medium]
```

Untuk setiap kemungkinan dari nama linguistic akan dikategorikan menjadi nano, micro, medium. Nilai dari linguistic per atribut akan diambil nilai maksimumnya. Kemudian dari nilai yang berada di output nya akan diambil nilai maksimalnya

Defuzzifikasi

```
def sugeno(inf):
    s = ((40*inf[0]) + (60*inf[1]) + (70*inf[2])) / (inf[0]+inf[1]+inf[2])
    return s
```

Metode yang digunakan untuk defuzzifikasi adalah Takagi-Sugeno dengan batas fungsi keanggotaan 40, 60, dan 70. Untuk mendapatkan hasilnya setiap nilai dari hasil inferensi dikalikan dengan batas fungsi keanggotaannya dan dijumlah kemudian dibagi dengan jumlah hasil inferensi

Mengurutkan data dari terbesar ke terkecil

```
def sorting(defuzzy,index):
    sort = [x for _, x in sorted(zip(defuzzy,index), reverse=True)]
    return sort
```

Kemudian setelah mendapatkan hasil defuzzifikasi setiap data, akan diurutkan dari yang terbesar sampai terkecil dan akan diambil 20 data terbaik

Menyimpan file ke CSV

```
def saveData(target):
    with open('chosen.csv', mode='w', newline='') as csv_file:
        csv_writer = csv.writer(csv_file)
        for index in target:
            csv_writer.writerow([index])
    return csv_writer
```

chosen.csv

	A
1	13
2	11
3	59
4	25
5	75
6	91
7	71
8	53
9	99
10	94
11	93
12	81
13	79
14	76
15	72
16	7
17	69
18	67
19	66
20	65