Penerapan *Fuzzy System* untuk Klasifikasi Status Indeks Massa Tubuh (BMI) Berdasarkan Berat Badan,Tinggi Badan, Persentase Lemak, dan Frekuensi Olahraga Menggunakan Dataset *Fitness Tracker*

Kelas :
IF-47-08

Disusun Oleh:

- Muhammad Fazlur Rahman 103012300405
- Bintang Saputra 103012300070

PROGRAM STUDI S1 INFORMATIKA

TELKOM UNIVERSITY

2024/2025

DAFTAR ISI

DAFTAR ISI **2**
BAB 1 PENDAHULUAN **3**
1.1 Latar Belakang 3
1.2 Rumusan Masalah 3
1.3 Solusi 4
1.4 Tujuan 4
BAB 2 PEMBAHASAN **5**
2.1 Paparan, Statistik, dan Sumber dari dataset 5
2.1.1 Dataset 5
2.1.2 Deskripsi Fitur 5
2.2 Paparan pre-processing dataset 6
2.2.1 Diagram Flow 6
2.2.2 Prapremrosesan Data 6
2.3 Implementasi Fuzzy 7
2.3.1 Input 7
2.3.2 Output 8
2.3.3 Fungsi Keanggotaan 8
2.3.4 Fuzzy Rule 12
2.3.5 Hasil 12
2.3.6 Evaluasi Fuzzy (Confusion Matrix) 13
2.4 Implementasi Random Forest 14
2.4.1 Persiapan Data 14
2.4.2 Pelatihan dan Cross-Validation 14
2.4.3 Simulasi Input dan Output 15
2.4.4 Hasil 15
2.4.5 Evaluasi (Confusion Matrix) 16
2.5 Perbandingan Fuzzy Logic dan Random Forest 17
BAB 3 PENUTUP **18**
3.1 Manfaat 18
3.2 Kesimpulan 18
3.3 Tautan 18
DAFTAR PUSTAKA **19**

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Kepuasan pelanggan merupakan salah satu faktor kunci dalam menentukan keberhasilan suatu bisnis. Data feedback dari pelanggan dapat memberikan informasi berharga terkait kualitas layanan dan produk yang diberikan oleh perusahaan. Namun, data feedback sering kali bersifat subjektif dan ambigu, sehingga pendekatan konvensional sulit untuk menanganinya secara optimal.

Metode Fuzzy Logic menawarkan solusi untuk mengatasi ketidakpastian dan ketidaktegasan dalam data subjektif seperti penilaian pelanggan. Dengan menerapkan sistem fuzzy, kita dapat mengubah data linguistik seperti "sangat puas", "puas", atau "tidak puas" menjadi nilai numerik untuk dianalisis secara kuantitatif dan mengambil keputusan yang lebih informatif.

1.2 Rumusan Masalah

- Apa saja faktor yang dapat mempengaruhi tingkat kepuasan pelanggan berdasarkan hasil fuzzy logic?
- Seberapa akurat hasil dari fuzzy logic dalam mengklasifikasikan kepuasan pelanggan?
- Bagaimana cara menerapkan fuzzy logic untuk mengklasifikasikan kepuasan pelanggan berdasarkan dari data feedback pelanggan?

1.3 Solusi

Solusi yang dilakukan adalah membangun sistem agar bisa mengatasi ketidakpastian dan ketidaktegasan dalam datasubjektif seperti penilaian pelanggan berbasis Fuzzy Logic yang dapat diakses dan dipahami oleh masyarakat umum. Sistem ini memanfaatkan data sederhana seperti kualitas produk, kualitas servis serta feedback score.

1.4 Tujuan

- Penerapan fuzzy logic dalam menganalisis data feedback pelanggan dan mengklasifikasikan kepuasan pelanggan.
- Mengidentifikasi pola-pola feedback yang berkorelasi dengan kepuasan pelanggan.

BAB 2 PEMBAHASAN

2.1 Paparan, Statistik, dan Sumber dari dataset

2.1.1 Dataset

Dataset yang kami gunakan dalam tugas besar ini adalah *feedback and satisfaction* yang kami dapatkan dari platform Kaggle feedback and satisfaction

2.1.2 Deskripsi Fitur

No	Nama Fitur	Deskripsi
1	Quality product	Kualitas produk
2	Quality service	Kualitas Servis
3	Feedback Score	Score umpanbalik
4	Satisfaction Score	Nilai kepuasaan(target output)

2.2 Paparan pre-processing dataset

2.2.1 Prapremrosesan Data

Melakukan penghapusan baris data yang terdapat nilai kosong, memperbaiki nilai yang salah.

# ProductQu...	# ServiceQu...	Feedback...	# Satisfactio...
5	8	Low	100.0
10	2	Medium	100.0
8	10	Medium	100.0
7	10	Low	100.0
6	4	Low	82.0
5	7	High	80.71
10	6	High	100.0
10	10	Medium	100.0
10	2	High	100.0
6	4	High	86.48
9	3	High	94.6
8	6	Medium	81.85
8	6	Medium	95.99
6	8	Medium	100.0
1	8	Medium	100.0
1	5	Medium	52.34
5	4	Medium	73.48

2.3 Implementasi Fuzzy

2.3.1 Input

1. **ProductQuality** (Skala 1-10):
    - Fungsi keanggotaan: Poor, Average, Good
2. **ServiceQuality** (Skala 1-10):
    - Fungsi keanggotaan: Poor, Average, Good
3. **FeedbackScore** (Kategorikal: Low, Medium, High):
    - Direpresentasikan sebagai nilai crisp.

2.3.2 Output

SatisfactionCategory: Low, Medium, High

2.3.3 Fungsi Keanggotaan

- **Satisfaction**
-

```python
# Kategorikan SatisfactionScore menjadi Low, Medium, High
# SatisfactionScore adalah target output
def categorize_satisfaction(score):
    if score <= 40:
        return 'Low'
    elif score <= 70:
        return 'Medium'
    else:
        return 'High'
```

- **Triangle**

```python
# Fungsi keanggotaan triangle
def triangular_mf(x, params):
    a, b, c = params
    if x <= a or x >= c:
        return 0.0
    elif a < x <= b:
        return (x - a) / (b - a)
    else:
        return (c - x) / (c - b)
```

✓ 0.0s

- **Fuzzy Function**

```python
# Fungsi fuzzifikasi untuk ProductQuality (pq = ProductQuality)
def fuzzify_pq(value):
    poor = triangular_mf(value, [1, 1, 5])
    average = triangular_mf(value, [3, 5, 7])
    good = triangular_mf(value, [5, 10, 10])
    return {'Poor': poor, 'Average': average, 'Good': good}

# Fungsi fuzzifikasi untuk ServiceQuality (sq = ServiceQuality)
def fuzzify_sq(value):
    poor = triangular_mf(value, [1, 1, 5])
    average = triangular_mf(value, [3, 5, 7])
    good = triangular_mf(value, [5, 10, 10])
    return {'Poor': poor, 'Average': average, 'Good': good}

# Fuzzifikasi untuk FeedbackScore
def fuzzify_fs(value):
    if value == 'Low':
        return {'Low': 1.0, 'Medium': 0.0, 'High': 0.0}
    elif value == 'Medium':
        return {'Low': 0.0, 'Medium': 1.0, 'High': 0.0}
    elif value == 'High':
        return {'Low': 0.0, 'Medium': 0.0, 'High': 1.0}
```

- **Inferensi madani**

```
# Inferensi Mamdani
def mamdani_inference(row):
    pq_val = row['ProductQuality']
    sq_val = row['ServiceQuality']
    fs_val = row['FeedbackScore']

    pq_fuzzy = fuzzify_pq(pq_val)
    sq_fuzzy = fuzzify_sq(sq_val)
    fs_fuzzy = fuzzify_fs(fs_val)

    output_strength = {'Low': 0.0, 'Medium': 0.0, 'High': 0.0}

    for rule in rules:
        pq_term = rule['pq']
        sq_term = rule['sq']
        fs_term = rule['fs']
        output_term = rule['output']

        strength = min(
            pq_fuzzy[pq_term],
            sq_fuzzy[sq_term],
            fs_fuzzy[fs_term]
        )

        if strength > output_strength[output_term]:
            output_strength[output_term] = strength

    # Defuzzifikasi dengan mean of maxima
    max_strength = max(output_strength.values())
    best_outputs = [k for k, v in output_strength.items() if v == max_strength]

    # Prioritas: High > Medium > Low
    if 'High' in best_outputs:
        return 'High'
    elif 'Medium' in best_outputs:
        return 'Medium'
    else:
        return 'Low'


# Penerapan inferensi mamdani pada data test
y_pred_mamdani = X_test.apply(mamdani_inference, axis=1)
```

- **Evaluate model**

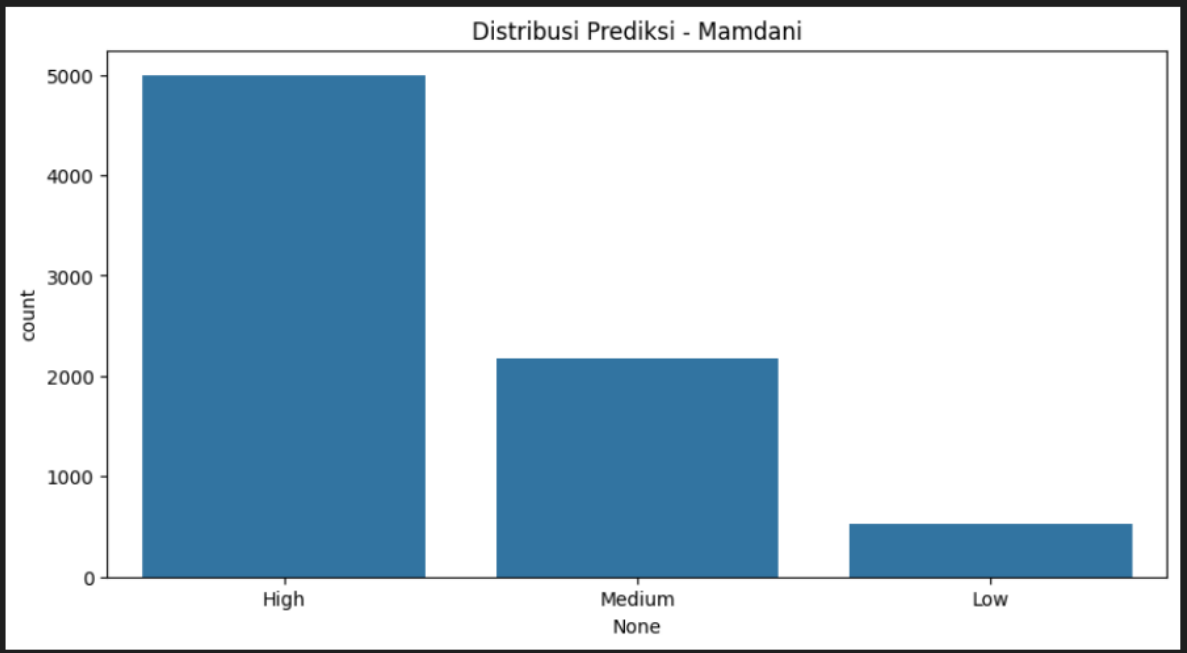
```python
def evaluate_model(y_true, y_pred, model_name):
    print(f"Evaluasi Model: {model_name}")
    accuracy = accuracy_score(y_true, y_pred)
    precision = precision_score(y_true, y_pred, average='weighted', zero_division=0)
    recall = recall_score(y_true, y_pred, average='weighted')
    f1 = f1_score(y_true, y_pred, average='weighted')

    print("Akurasi:", accuracy)
    print("Presisi:", precision)
    print("Recall:", recall)
    print("F1-Score:", f1)
    print("\\n")

    return {
        'Model': model_name,
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1': f1
    }
```

2.3.4 Fuzzy Rule

```
rules = [
    # ProductQuality Poor
    {'pq': 'Poor', 'sq': 'Poor', 'fs': 'Low', 'output': 'Low'},
    {'pq': 'Poor', 'sq': 'Poor', 'fs': 'Medium', 'output': 'Low'},
    {'pq': 'Poor', 'sq': 'Poor', 'fs': 'High', 'output': 'Medium'},

    {'pq': 'Poor', 'sq': 'Average', 'fs': 'Low', 'output': 'Low'},
    {'pq': 'Poor', 'sq': 'Average', 'fs': 'Medium', 'output': 'Medium'},
    {'pq': 'Poor', 'sq': 'Average', 'fs': 'High', 'output': 'Medium'},

    {'pq': 'Poor', 'sq': 'Good', 'fs': 'Low', 'output': 'Medium'},
    {'pq': 'Poor', 'sq': 'Good', 'fs': 'Medium', 'output': 'Medium'},
    {'pq': 'Poor', 'sq': 'Good', 'fs': 'High', 'output': 'High'},

    # ProductQuality Average
    {'pq': 'Average', 'sq': 'Poor', 'fs': 'Low', 'output': 'Low'},
    {'pq': 'Average', 'sq': 'Poor', 'fs': 'Medium', 'output': 'Medium'},
    {'pq': 'Average', 'sq': 'Poor', 'fs': 'High', 'output': 'Medium'},

    {'pq': 'Average', 'sq': 'Average', 'fs': 'Low', 'output': 'Medium'},
    {'pq': 'Average', 'sq': 'Average', 'fs': 'Medium', 'output': 'Medium'},
    {'pq': 'Average', 'sq': 'Average', 'fs': 'High', 'output': 'High'},

    {'pq': 'Average', 'sq': 'Good', 'fs': 'Low', 'output': 'Medium'},
    {'pq': 'Average', 'sq': 'Good', 'fs': 'Medium', 'output': 'High'},
    {'pq': 'Average', 'sq': 'Good', 'fs': 'High', 'output': 'High'},

    # ProductQuality Good
    {'pq': 'Good', 'sq': 'Poor', 'fs': 'Low', 'output': 'Medium'},
    {'pq': 'Good', 'sq': 'Poor', 'fs': 'Medium', 'output': 'Medium'},
    {'pq': 'Good', 'sq': 'Poor', 'fs': 'High', 'output': 'High'},

    {'pq': 'Good', 'sq': 'Average', 'fs': 'Low', 'output': 'Medium'},
    {'pq': 'Good', 'sq': 'Average', 'fs': 'Medium', 'output': 'High'},
    {'pq': 'Good', 'sq': 'Average', 'fs': 'High', 'output': 'High'},

    {'pq': 'Good', 'sq': 'Good', 'fs': 'Low', 'output': 'High'},
    {'pq': 'Good', 'sq': 'Good', 'fs': 'Medium', 'output': 'High'},
    {'pq': 'Good', 'sq': 'Good', 'fs': 'High', 'output': 'High'}
]
```

2.3.5 Hasil

\n

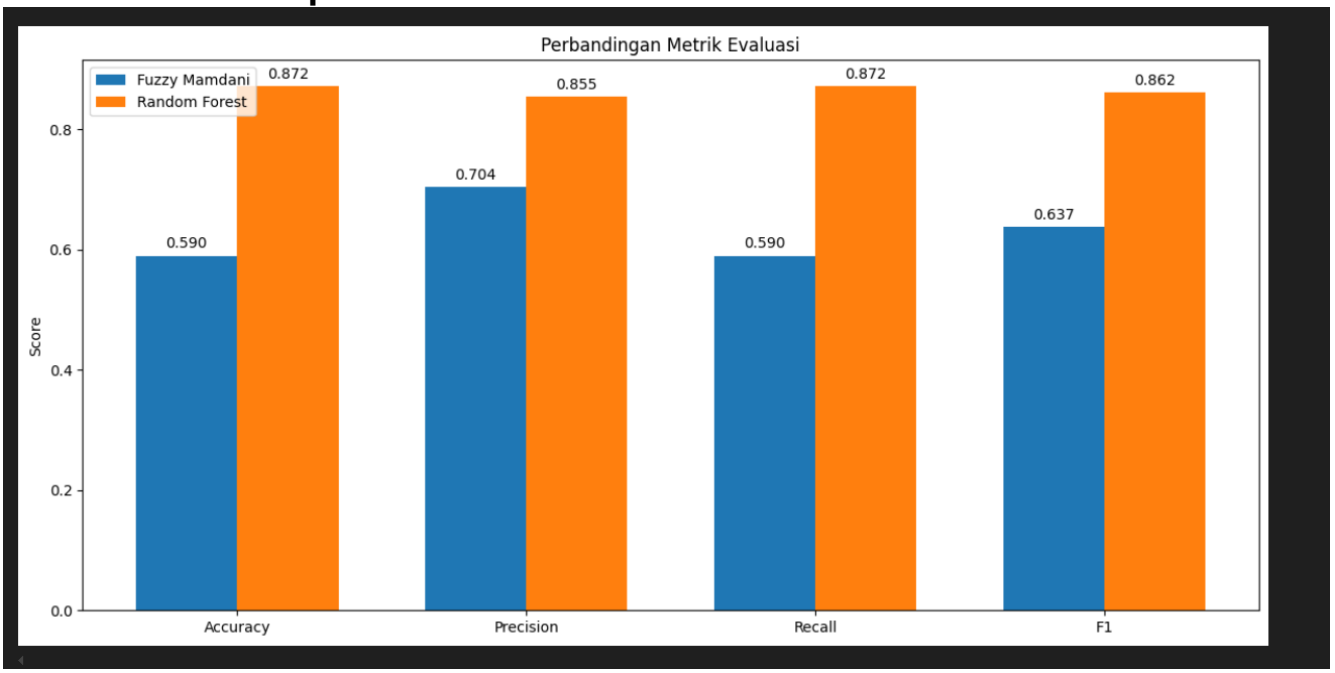

Distribusi Prediksi - Mamdani
count
5000
4000
3000
2000
1000
0
High
Medium
Low
None

2.3.6 Evaluasi Fuzzy (Confusion Matrix)

```
Evaluasi Model: Fuzzy Mamdani
Akurasi: 0.5900637274027832
Presisi: 0.7040223687452287
Recall: 0.5900637274027832
F1-Score: 0.6374913436702027
```

2.4 Implementasi Random Forest

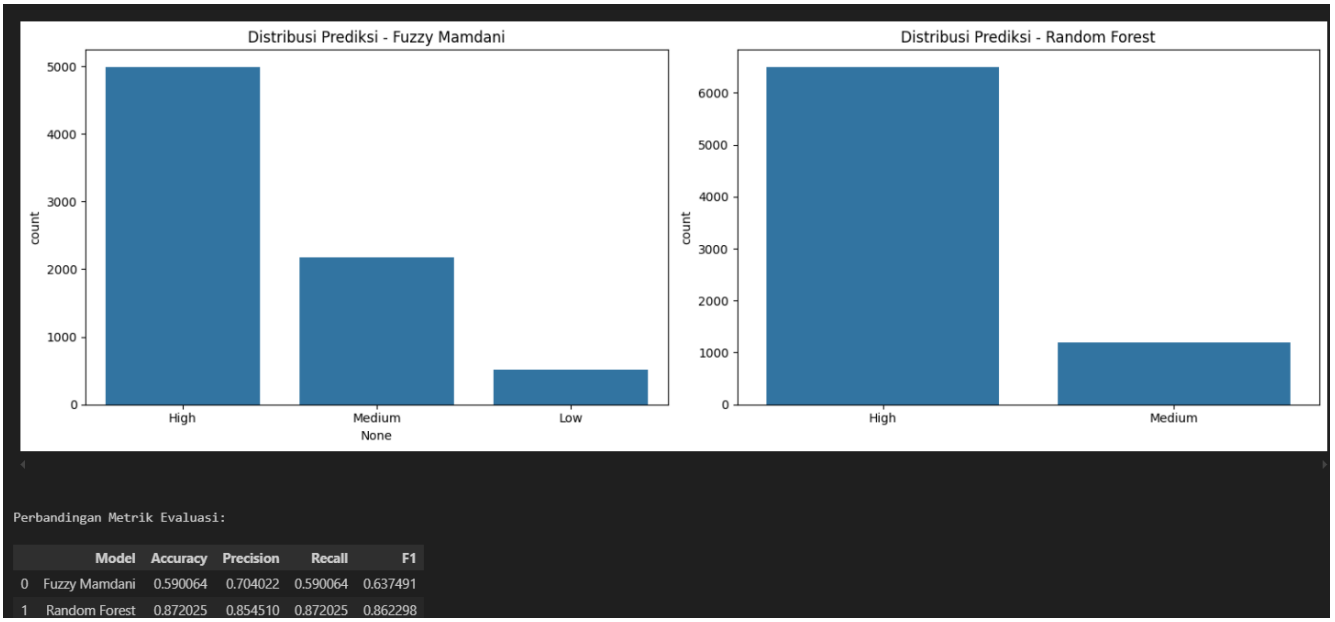
2.4.1 Persiapan Data



2.4.2 Pelatihan dan Cross-Validation

```python
# melatih model random forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train_encoded)

# Memprediksi dengan random forest
y_pred_rf_encoded = rf_model.predict(X_test)
y_pred_rf = le.inverse_transform(y_pred_rf_encoded)

# Evaluasi Random Forest
rf_metrics = evaluate_model(y_test, y_pred_rf, "Random Forest")
```

2.4.3 Hasil

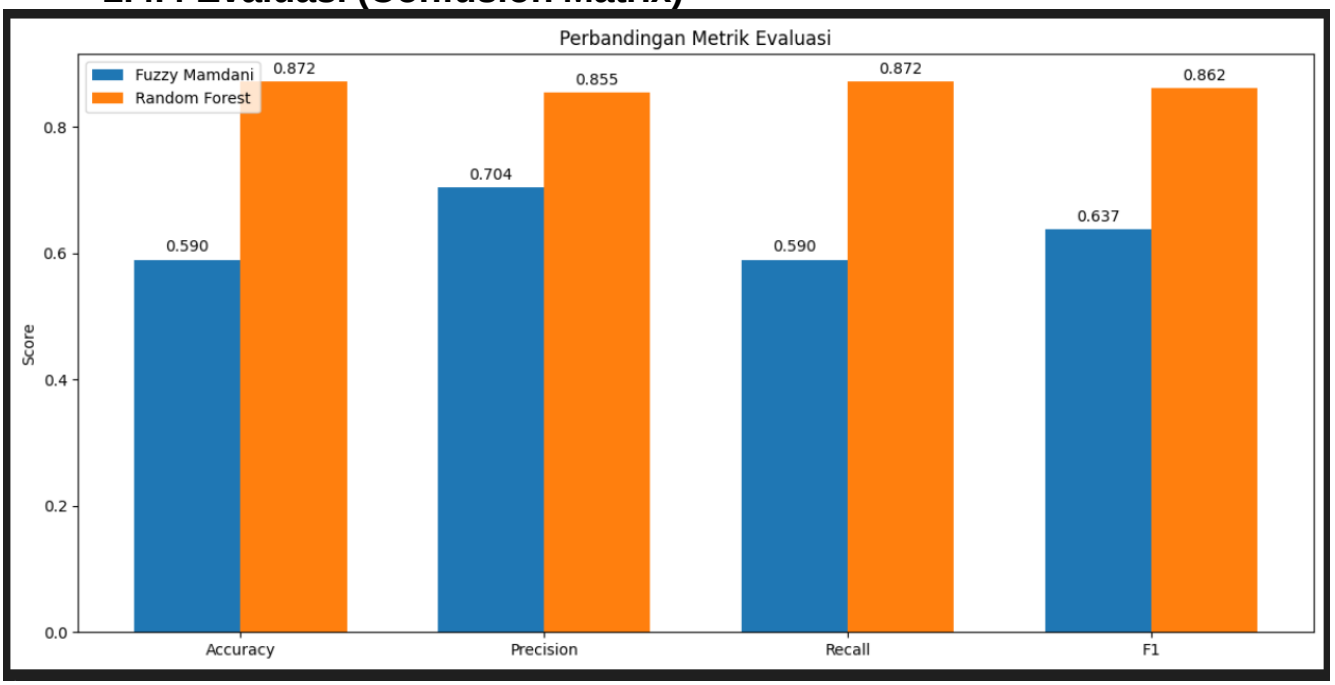


Perbandingan Metrik Evaluasi:

	Model	Accuracy	Precision	Recall	F1
0	Fuzzy Mamdani	0.590064	0.704022	0.590064	0.637491
1	Random Forest	0.872025	0.854510	0.872025	0.862298

- Random Forest menunjukkan kinerja superior karena kemampuannya mempelajari pola kompleks dalam data.
- **Fuzzy Mamdani** memiliki presisi yang relatif tinggi, namun recall dan akurasi lebih rendah. Hal ini disebabkan oleh tantangan dalam mendefinisikan aturan fuzzy yang optimal.
- Meskipun akurasi fuzzy lebih rendah, sistem ini memberikan keunggulan dalam interpretasi dan transparansi aturan.

2.4.4 Evaluasi (Confusion Matrix)



2.5 Perbandingan Fuzzy Logic dan Random Forest

- **Random Forest**
  Model Random Forest dilatih untuk memprediksi kategori kepuasan pelanggan berdasarkan fitur ProductQuality, ServiceQuality, dan FeedbackScore. Dengan kemampuan menangkap hubungan kompleks antara fitur dan target, model ini mencapai akurasi tinggi (87.2%) karena:

1. Memanfaatkan pola tersembunyi dalam data secara optimal
2. Robust terhadap variasi data
3. Menggunakan ensemble learning untuk mengurangi overfitting
4. Secara otomatis menangkap hubungan non-linear antar variabel

- **Fuzzy Logic**

- Fuzzy.Mamdani
  Sistem berbasis aturan ini dirancang untuk meniru penalaran manusia dengan:
  1. Mendefinisikan variabel linguistik (Poor, Average, Good)
  2. Membangun rulebase berdasarkan pengetahuan domain (27 rules)
  3. Menggunakan inferensi Mamdani untuk penalaran fuzzy
  4. Hasil akhir menunjukkan akurasi lebih rendah (59.0%) karena:
     - Ketergantungan pada aturan eksplisit yang mungkin tidak menangkap semua kompleksitas data
     - Sulitnya menentukan parameter fungsi keanggotaan yang optimal
     - Terbatasnya kemampuan adaptasi terhadap pola data baru

BAB 3 PENUTUP

3.1 Manfaat

Akurasi:

- Random Forest unggul 47.7% dalam akurasi
- Mampu menangkap pola kompleks yang terlewat oleh aturan manual

Interpretabilitas:

- Fuzzy Mamdani lebih mudah diinterpretasi (aturan linguistik transparan)
- Random Forest beroperasi seperti "blackbox" meski akurat

Adaptabilitas:

- Random Forest belajar otomatis dari data
- Fuzzy Mamdani membutuhkan penyetelan manual parameter dan aturan

Kinerja Kelas Minoritas:

- Random Forest lebih konsisten di semua kategori (Low/Medium/High)
- Fuzzy Mamdani cenderung under-prediction untuk kategori High

3.2 Kesimpulan

Gunakan **Random Forest** untuk sistem yang memprioritaskan akurasi prediksi

- Gunakan **Fuzzy Mamdani** ketika:
    - Interpretasi proses keputusan penting
    - Domain knowledge eksperts tersedia
    - Fleksibilitas penyesuaian aturan diperlukan
- Hybrid system: Gabungkan kekuatan kedua metode dengan menggunakan output Random Forest sebagai input sistem fuzzy untuk penjelasan yang lebih interpretable

3.3 Tautan

Slide Presentation :
https://www.canva.com/design/DAGpHBM_-Kg/wPMIkkQkX0ql8kxNBi8EgQ/edit?utm_content=DAGpHBM_-Kg&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton