

PERBANDINGAN FUZZY LOGIC DAN RANDOM FOREST DALAM PREDIKSI TINGKAT OBESITAS

**NEVAN NABIIL FIRMANSYAH H - 103012300273
DAVINO PUTRA ARRAYA -103012300522**



Latar Belakang

Obesitas merupakan salah satu permasalahan kesehatan masyarakat yang semakin meningkat secara global. Kondisi ini dapat memicu berbagai komplikasi serius seperti diabetes tipe 2, penyakit jantung, dan hipertensi. Oleh karena itu, diperlukan sistem yang mampu memprediksi risiko obesitas berdasarkan karakteristik individu dan pola hidup sehari-hari untuk mendukung tindakan preventif.

Rumusan Masalah

- 1.Bagaimana membangun sistem prediksi obesitas menggunakan metode Fuzzy Logic Mamdani?
- 2.Bagaimana performa Random Forest dalam memprediksi kategori obesitas dibandingkan dengan metode fuzzy?
- 3.Algoritma manakah yang memberikan hasil paling optimal dalam prediksi obesitas berdasarkan fitur fisik dan gaya hidup?

Tujuan Penelitian

1. Mengimplementasikan sistem prediksi obesitas menggunakan metode Fuzzy Logic Mamdani, serta Random Forest.
2. Membandingkan performa ketiga metode tersebut menggunakan metrik evaluasi yang sesuai.
3. Menganalisis kekuatan dan kelemahan pendekatan berbasis aturan dan pendekatan statistik dalam klasifikasi multikelas.



Dataset “Obesity Prediction”

Deskripsi Dataset

Dataset yang digunakan berjudul "Obesity Prediction", yang berisi data demografis, atribut fisik, dan gaya hidup individu. Dataset ini bersifat publik dan tersedia dalam format tabular. Label target yang digunakan adalah Obesity Category, yaitu kategori tingkat obesitas individu berdasarkan standar BMI.

Alasan Pemilihan

Dataset Obesity Prediction dipilih karena memiliki relevansi tinggi dengan isu kesehatan masyarakat yang nyata dan mendesak. Obesitas merupakan kondisi yang dapat dicegah, namun sering kali tidak terdeteksi secara dini. Dataset ini memungkinkan prediksi tingkat obesitas berdasarkan atribut penting seperti tinggi badan, umur, jenis kelamin, berat badan, BMI, dan tingkat aktivitas fisik seseorang.

Fitur

- Height : Tinggi Badan Pasien (meter)
- Age : Umur Pasien (tahun)
- Gender : Jenis kelamin pasien. Fitur ini dikodekan secara numerik, dengan nilai 0 untuk perempuan dan 1 untuk laki-laki.
- Weight : Berat Badan Pasien (kg)
- BMI : Ukuran yang digunakan untuk mengevaluasi apakah berat badan seseorang proporsional dengan tinggi badannya
- PhysicalActivitiesLevel : Variabel numerik yang mengukur tingkat aktivitas fisik individu, dengan nilai berkisar dari 1 (rendah) hingga 4 (tinggi)
- ObesityCategory : Kategori obesitas (Underweight, Normal, Overweight, Obesity)

METODOLOGI

Pra-pemrosesan Data

Pemeriksaan dan penanganan missing values

Untuk memastikan bahwa tidak ada nilai kosong (missing values) dalam dataset, langkah pembersihan dilakukan dengan langsung menghapus seluruh baris yang mengandung nilai NaN.

```
# Tangani missing values (hapus baris yang mengandung NaN)
df = df.dropna().reset_index(drop=True)
```

Pemeriksaan dan penanganan missing values

Deteksi outlier dilakukan menggunakan metode Z-score, dan nilai Weight dengan Z-score di atas 3 dianggap sebagai outlier ekstrem dan dihapus dari dataset. Ini dilakukan untuk menjaga kualitas data dan mencegah bias atau gangguan yang ditimbulkan oleh nilai ekstrem terhadap model prediksi.

```
# Deteksi dan hapus outlier dari 'Weight'
z_scores = np.abs(stats.zscore(df['Weight']))
df_clean = df[z_scores < 3].copy()
```

Normalisasi Fitur Numerik

Fitur numerik seperti Age, Weight, Height, BMI, dan PhysicalActivityLevel dinormalisasi menggunakan metode StandardScaler dari library scikit-learn. Metode ini mengubah setiap nilai fitur menjadi distribusi standar dengan rata-rata 0 dan standar deviasi 1. Normalisasi ini penting khususnya untuk model Logistic Regression yang sensitif terhadap skala antar fitur.

```
# Normalisasi fitur numerik
numerical_cols = ['Age', 'Weight', 'Height', 'BMI', 'PhysicalActivityLevel']
scaler = StandardScaler()
df_clean[numerical_cols] = scaler.fit_transform(df_clean[numerical_cols])
```



Encoding Fitur Kategorikal

Fitur Gender, yang semula berupa data kategorikal (Male/Female), dikonversi menjadi data numerik menggunakan metode Label Encoding, dengan nilai 0 untuk "Female" dan 1 untuk "Male". Proses ini dilakukan agar model klasifikasi dapat memproses fitur ini secara numerik.

```
# Encode 'Gender' (Male/Female -> 0/1)
gender_encoder = LabelEncoder()
df_clean['Gender'] = gender_encoder.fit_transform(df_clean['Gender'])
```

Encoding Target Label

Kolom ObesityCategory yang merupakan label target dikonversi menjadi format numerik menggunakan LabelEncoder. Kategori seperti "Normal weight", "Obesity", dan lainnya diubah menjadi nilai numerik agar bisa diproses sebagai target klasifikasi multi-kelas.

```
# Encode 'ObesityCategory' sebagai target (misalnya: Normal weight -> 0, Obese -> 1, dst)
label_encoder = LabelEncoder()
df_clean['ObesityCategory'] = label_encoder.fit_transform(df_clean['ObesityCategory'])
```

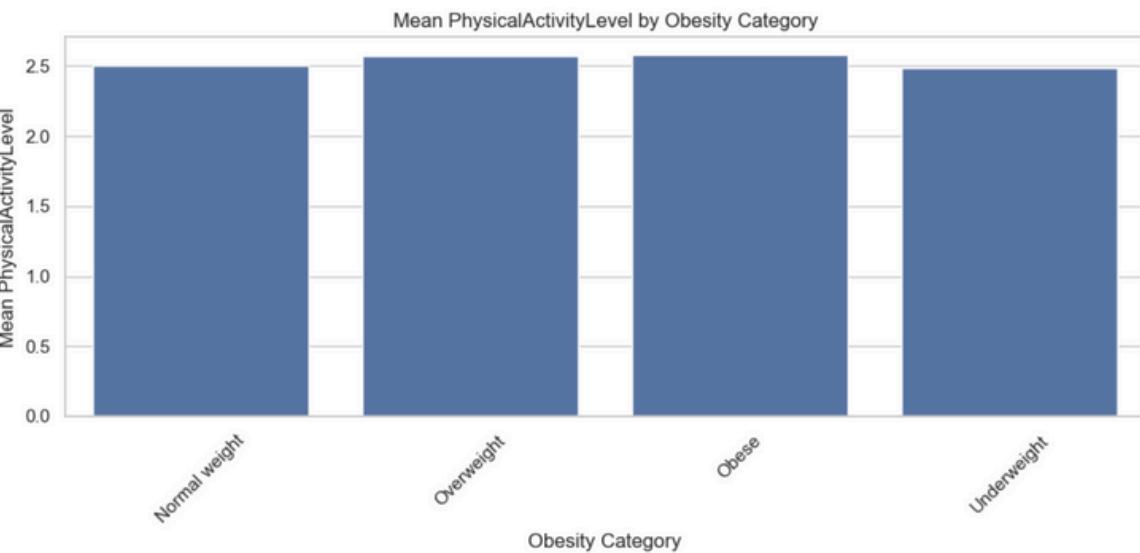
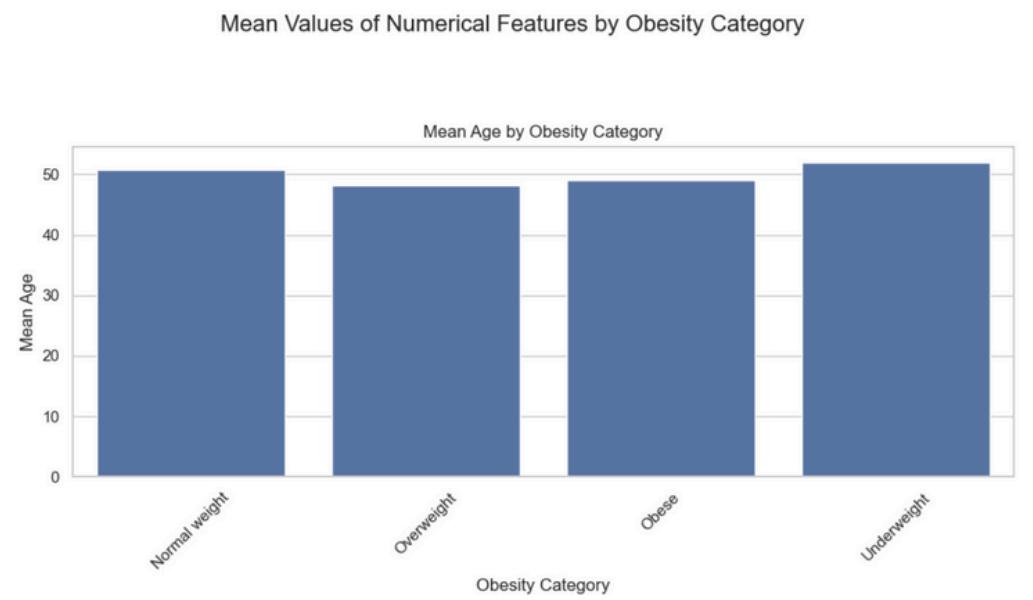
Penyimpanan Dataset Bersih

Dataset yang telah melalui proses pembersihan, normalisasi, dan encoding kemudian disimpan ke dalam file baru bernama "obesity_data_processed_final_1.csv", agar dapat digunakan kembali dalam implementasi model Logistic Regression, Fuzzy Logic, dan model Hybrid.

```
# Simpan ke file CSV
df_clean.to_csv("obesity_data_processed_final_1.csv", index=False)
```

Exploratory Data Analysis

Eksplorasi analisis dataset ini dilakukan untuk memahami distribusi dan hubungan antar fitur dalam dataset terhadap label. Fokus utama diberikan pada fitur-fitur yang sudah dipilih menjadi fitur utama, yaitu “Mean Values of Numerical Features by Obesity Category”.



Implementasi Algoritma - I

• A. Penentuan Fuzzy Variables dan Membership Function

Fitur yang digunakan sebagai input (antecedent) dalam sistem fuzzy mencakup:

- Height
- Weight
- BMI (Body Mass Index)
- Physical Activity Level
- Age
- Gender

Masing-masing fitur direpresentasikan sebagai **fuzzy variable (antecedent)**, sedangkan label target (kategori obesitas) direpresentasikan sebagai **consequent (Risk)**.

```
Weight = ctrl.Antecedent(Weight_uni, 'Weight')
Height = ctrl.Antecedent(Height_uni, 'Height')
BMI    = ctrl.Antecedent(BMI_uni, 'BMI')
Activity = ctrl.Antecedent(Activity_uni, 'Activity')
Age     = ctrl.Antecedent(Age_uni, 'Age')
Gender   = ctrl.Antecedent(Gender_uni, 'Gender')
Risk    = ctrl.Consequent(Risk_uni, 'Risk')
```

Untuk fitur-fitur numerik seperti BMI, PhysicalActivityLevel, Age, Weight, dan Height, masing-masing dibagi menjadi tiga kategori linguistik menggunakan fungsi keanggotaan segitiga (triangular membership function / trimf) yang ditentukan berdasarkan nilai kuartil (25%, 50%, dan 75%) dari data aktual.

```
# Quantile-based membership
def add_mfs(var, q_vals, names):
    lo, hi = var.universe.min(), var.universe.max()
    a,b,c = sorted([float(x) for x in q_vals])
    var[names[0]] = fuzz.trimf(var.universe, [lo, a, b])
    var[names[1]] = fuzz.trimf(var.universe, [a, b, c])
    var[names[2]] = fuzz.trimf(var.universe, [b, c, hi])

Q = {c: df[c].quantile([.25, .5, .75]).values for c in [
    'BMI', 'PhysicalActivityLevel', 'Age', 'Weight', 'Height'
]}
```

Fuzzy Logic (Mamdani)

Model Fuzzy Logic dikembangkan sebagai salah satu pendekatan utama dalam tugas besar ini untuk memprediksi tingkat obesitas berdasarkan atribut fisik dan gaya hidup. Pendekatan fuzzy dipilih karena kemampuannya dalam menangani ketidakpastian, ambiguitas, dan data linguistik, terutama pada konteks seperti kesehatan yang seringkali memiliki batas kategori yang tidak tegas.

Kemudian, fungsi ini diterapkan pada masing-masing variabel:

- BMI: low, medium, high
- Physical Activity: poor, fair, good
- Age: young, adult, senior
- Gender: female, male
- Weight: light, medium, heavy
- Height: short, average, tall

```
add_mfs(BMI, Q['BMI'], ['low', 'medium', 'high'])
add_mfs(Activity, Q['PhysicalActivityLevel'], ['poor', 'fair', 'good'])
add_mfs(Age, Q['Age'], ['young', 'adult', 'senior'])
add_mfs(Weight, Q['Weight'], ['light', 'medium', 'heavy'])
add_mfs(Height, Q['Height'], ['short', 'average', 'tall'])

# Membership function manual
Gender['female'] = fuzz.trimf(Gender.universe, [0, 0, 1])
Gender['male']   = fuzz.trimf(Gender.universe, [0, 1, 1])
```

Sedangkan untuk consequent Risk, ditentukan empat kategori:

- under
- normal
- over
- Obese

```
# Consequent MFs
Risk['under'] = fuzz.trimf(Risk.universe, [0,0,1])
Risk['norm']  = fuzz.trimf(Risk.universe, [0,1,2])
Risk['over']  = fuzz.trimf(Risk.universe, [1,2,3])
Risk['obese'] = fuzz.trimf(Risk.universe, [2,3,3])
```

Implementasi Algoritma - II

• B.Penyusunan Aturan (Fuzzy Rules)

Salah satu kekuatan utama dalam sistem Fuzzy Logic adalah kemampuan untuk membuat aturan berbasis logika manusia. Dalam implementasi ini, penyusunan aturan dilakukan berdasarkan kombinasi dari variabel-variabel input (BMI, Activity, Age, Gender, Weight, dan Height) terhadap variabel output Risk.

- Aturan Dasar: BMI vs Physical Activity

Aturan utama disusun menggunakan kombinasi antara BMI dan PhysicalActivityLevel, yang dianggap sebagai dua indikator paling langsung terhadap risiko obesitas. Aturan ini disusun dalam bentuk grid.

```
# Comprehensive rule grid for BMI vs Activity
rule_grid = [
    (BMI['low'], Activity['poor'], 'under'),
    (BMI['low'], Activity['fair'], 'under'),
    (BMI['low'], Activity['good'], 'under'),

    (BMI['medium'], Activity['poor'], 'over'),
    (BMI['medium'], Activity['fair'], 'over'),
    (BMI['medium'], Activity['good'], 'norm'),

    (BMI['high'], Activity['poor'], 'obese'),
    (BMI['high'], Activity['fair'], 'obese'),
    (BMI['high'], Activity['good'], 'over'),
]

rules = [ctrl.Rule(cond1 & cond2, Risk[label]) for cond1,cond2,label in rule_grid]
```

- Aturan Penyesuaian Usia (Age Adjustment Rules)

Untuk memperhitungkan pengaruh usia terhadap risiko obesitas, aturan tambahan dibuat berdasarkan kategori usia "senior":

```
# Age adjustment rules
rules += [
    ctrl.Rule(Age['senior'] & BMI['medium'] & Activity['fair'], Risk['over']),
    ctrl.Rule(Age['senior'] & BMI['high'], Risk['obese']),
]
```

- Aturan Berbasis Gender

Fitur Gender merupakan salah satu variabel penting yang turut memengaruhi tingkat risiko obesitas. Oleh karena itu, sistem fuzzy ini menyertakan aturan yang mempertimbangkan perbedaan risiko berdasarkan jenis kelamin.

```
# Gender-based rules
rules += [
    ctrl.Rule(Gender['male'] & BMI['high'] & Activity['poor'], Risk['obese']),
    ctrl.Rule(Gender['female'] & BMI['medium'] & Activity['good'], Risk['norm']),
]
```

- Aturan Berdasarkan Weight dan Height

Fitur Weight dan Height juga digunakan dalam penyusunan aturan karena keduanya berperan penting dalam menentukan proporsi tubuh seseorang.

```
# Weight & Height based rules
rules += [
    ctrl.Rule(Weight['heavy'] & Height['short'], Risk['obese']),
    ctrl.Rule(Weight['light'] & Height['tall'], Risk['under']),
    ctrl.Rule(Weight['medium'] & Height['average'], Risk['norm']),
]
```

Implementasi Algoritma - III

• C.Inferensi dan Predksi

Setelah semua aturan fuzzy disusun, proses selanjutnya adalah melakukan inferensi (penyimpulan) dan prediksi terhadap data.

Setiap baris data pada dataset digunakan sebagai input ke dalam sistem fuzzy yang telah dibuat. Nilai-nilai fitur seperti **BMI**, **PhysicalActivityLevel**, **Age**,

Gender, **Weight**, dan **Height** dimasukkan ke dalam sistem, lalu dilakukan proses inferensi fuzzy menggunakan metode Mamdani.

Setelah inferensi selesai, dilakukan proses defuzzifikasi untuk mendapatkan output dalam bentuk nilai numerik, yang kemudian dibulatkan ke dalam salah satu kategori kelas **ObesityCategory** (yaitu: 0 = Underweight, 1 = Normal, 2 = Overweight, 3 = Obesity).

```
y_pred = []
for row in df.itertuples(index=False):
    sim = ctrl.ControlSystemSimulation(system)
    sim.input['BMI']      = row.BMI
    sim.input['Activity'] = row.PhysicalActivityLevel
    sim.input['Age']       = row.Age
    sim.input['Gender']   = row.Gender
    sim.input['Weight']   = row.Weight
    sim.input['Height']   = row.Height
    sim.compute()
    risk = sim.output.get('Risk', np.nan)
    if np.isnan(risk): # fallback if somehow none
        risk = 1 # default to 'Normal'
    y_pred.append(int(np.clip(round(risk), 0, 3)))

print('Predictions done:', len(y_pred))
```

Output prediksi ini akan digunakan kembali untuk melakukan perbandingan dengan hasil performa model Logistic Regression dan Hybrid Model

```
report = classification_report(df[y_col], y_pred, output_dict=True)
report_df = pd.DataFrame(report).transpose()
print(report_df)
ConfusionMatrixDisplay(cm, display_labels=['Under', 'Normal', 'Over', 'Obese']).plot(cmap='Blues', xticks_rotation=45)
plt.show()
```

Dengan proses ini, seluruh data berhasil diprediksi oleh sistem fuzzy, dan hasilnya siap digunakan dalam tahap evaluasi performa model.

Implementasi Algoritma - IV

Random Forest

Random Forest digunakan sebagai pendekatan berbasis ensemble learning untuk memprediksi kategori obesitas. Model ini dipilih karena kemampuannya dalam menangani klasifikasi multi-kelas dengan akurasi tinggi, tahan terhadap overfitting, serta efektif dalam mengolah data tabular dengan fitur yang kompleks.

• A.Pemilihan Fitur

Fitur yang digunakan untuk model ini adalah:

- **Age** (umur)
- **Gender** (jenis kelamin)
- **Weight** (berat badan)
- **Height** (tinggi badan)
- **BMI** (Body Mass Index)
- **PhysicalActivityLevel** (tingkat aktivitas fisik)

Fitur-fitur tersebut dipilih karena memiliki hubungan yang kuat terhadap status obesitas seseorang.

Target label (**y**) adalah **ObesityCategory**, yang merupakan klasifikasi multikelas dengan label:

- 0 = Underweight
- 1 = Normal
- 2 = Overweight
- 3 = Obesity

• B.Pembagian Data

Dataset yang telah dipra-pemrosesan dibagi menjadi dua bagian:

- Data latih (training) sebesar 80%
- Data uji (testing) sebesar 20%

Pembagian dilakukan secara stratified untuk memastikan distribusi kelas tetap seimbang.

```
# fitur (x) Label (y)
X = df[['Age', 'Gender', 'Weight', 'Height', 'BMI', 'PhysicalActivityLevel']]
y = df['ObesityCategory']

# split data: training dan testing (80:20)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

• C.Pelatihan Model

Model Random Forest dilatih menggunakan library sklearn dengan pengaturan `random_state=7` untuk memastikan konsistensi hasil pelatihan saat dijalankan berkali-kali. Dengan menggunakan data latih (`X_train`, `y_train`), model belajar mengenali pola dari fitur-fitur yang ada untuk memprediksi kategori obesitas secara efektif.

Latih model

```
model = RandomForestClassifier(random_state=7)
model.fit(X_train, y_train)
```

• D.Prediksi dan Simpan Hasil

Setelah model dilatih, dilakukan prediksi pada data uji dan hasilnya disimpan ke model Random Forest:

```
# simpan hasil
logreg_df = pd.DataFrame({
    'Actual': y_test.values,
    'Predicted_LogReg': y_pred
})
logreg_df.to_csv("Randomf_predictions.csv", index=False)
```

Model Random Forest ini akan dibandingkan performanya dengan model **Fuzzy Logic** pada bagian evaluasi.

Evaluasi Model - I

Evaluasi dilakukan untuk menilai performa dari masing-masing model dalam memprediksi kategori obesitas. Tiga model yang dievaluasi yaitu:

1. Random Forest
2. Fuzzy Logic Mamdani

Metrik evaluasi yang digunakan meliputi Accuracy, Precision, Recall, F1-score, dan visualisasi Confusion Matrix.

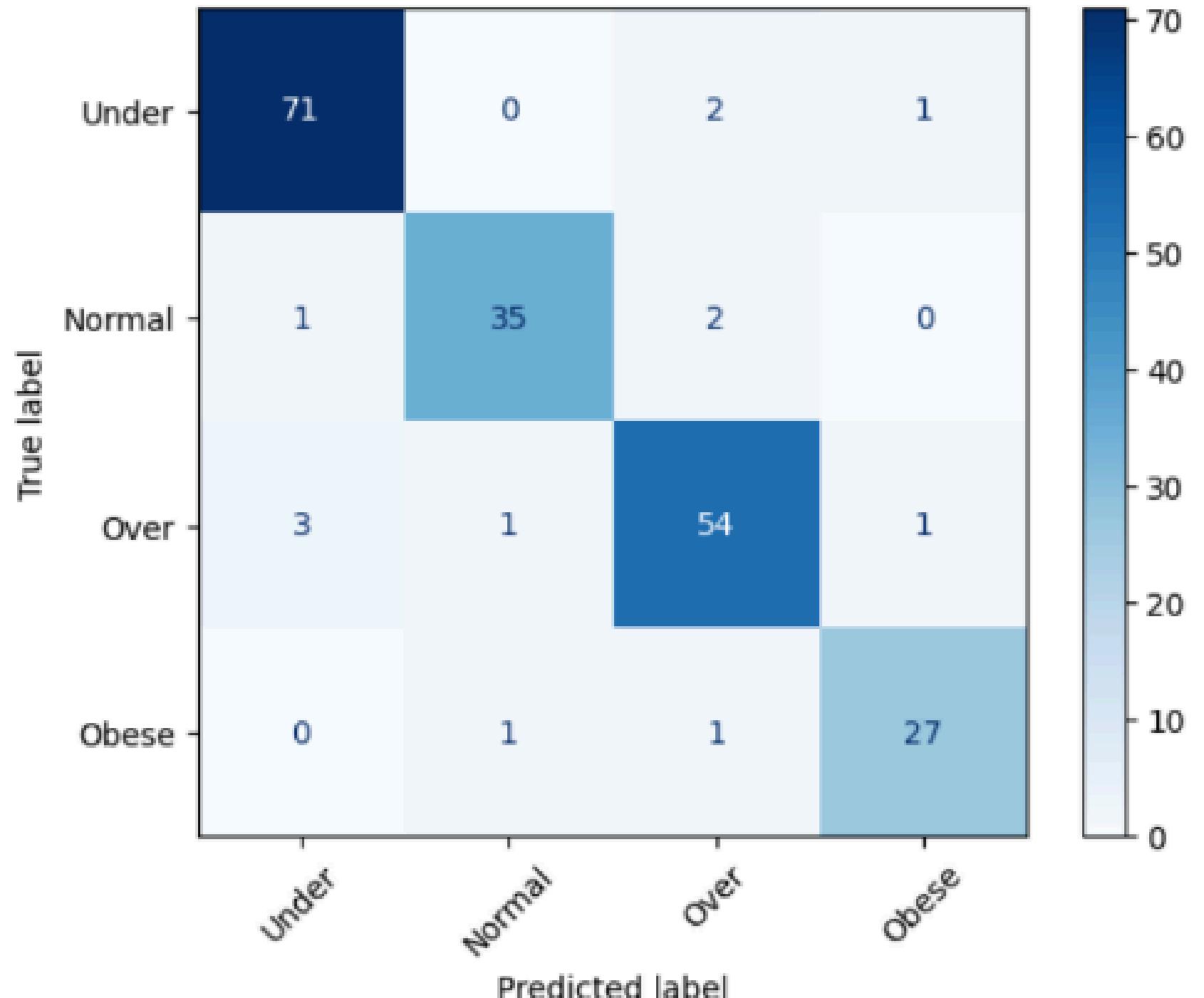
Evaluasi Model - II

Evaluasi model Rain Forest

Random Forest digunakan sebagai pendekatan berbasis ensemble learning untuk memprediksi kategori obesitas. Model ini dipilih karena kemampuannya dalam menangani klasifikasi multi-kelas dengan akurasi tinggi, tahan terhadap overfitting, serta efektif dalam mengolah data tabular dengan fitur yang kompleks.

Classification Report:

	precision	recall	f1-score	support
0	0.946667	0.959459	0.953020	74.000
1	0.945946	0.921053	0.933333	38.000
2	0.915254	0.915254	0.915254	59.000
3	0.931034	0.931034	0.931034	29.000
accuracy	0.935000	0.935000	0.935000	200.000
macro avg	0.934725	0.931700	0.933161	200.000
weighted avg	0.934996	0.935000	0.934951	200.000



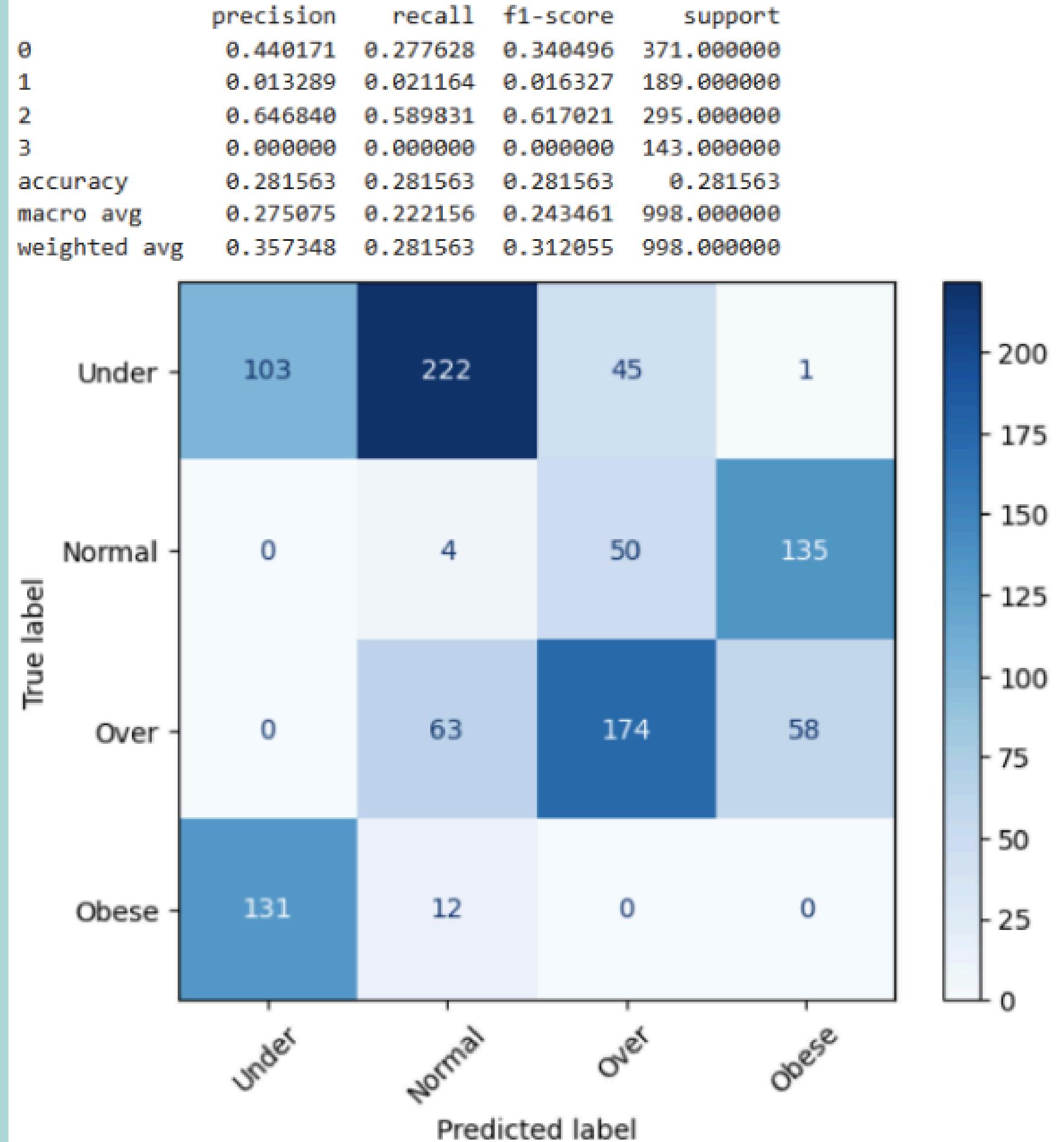
Evaluasi Model - III

Evaluasi model Fuzzy Logic Mamdani

Model fuzzy menggunakan seluruh dataset (998 data) karena tidak melakukan split data.

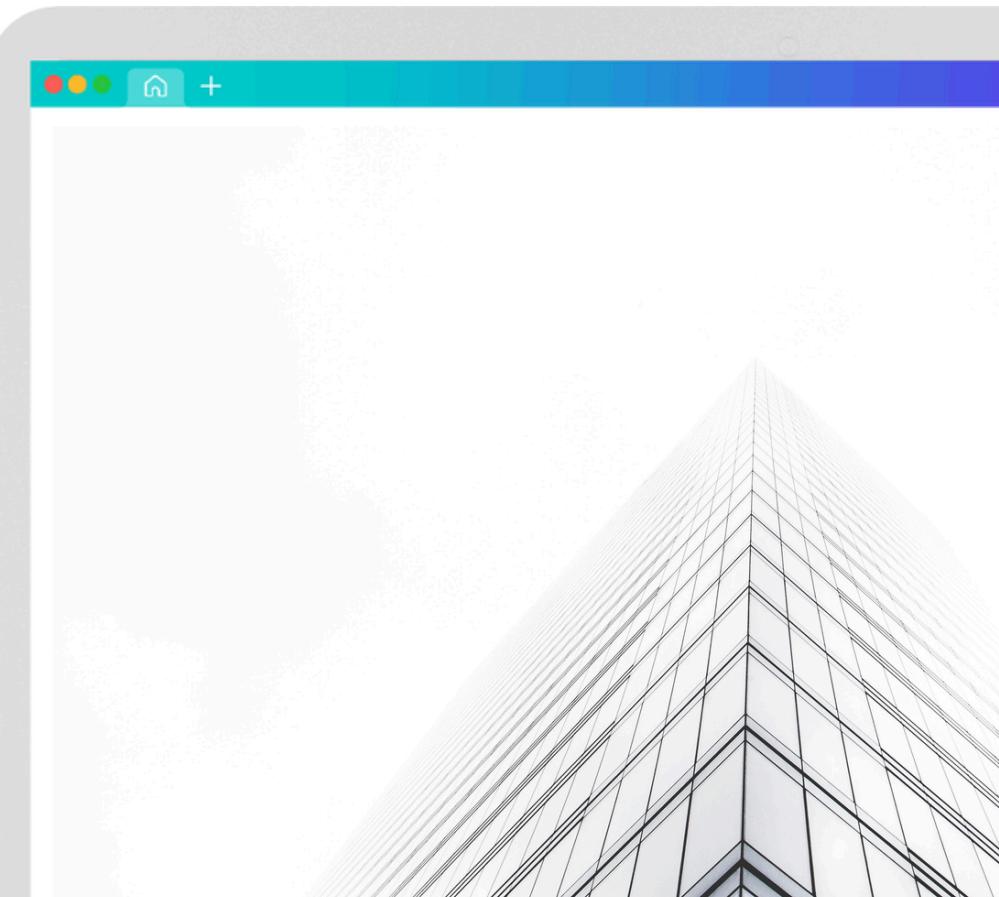
- Accuracy: 28.2%
- F1-score macro avg: 0.243
- Kelas Obese dan Normal sangat sulit diprediksi
- Banyak kesalahan klasifikasi antar kelas

Hasil ini menunjukkan bahwa performa fuzzy sangat bergantung pada kualitas aturan dan desain sistem.



Kesimpulan

Random Forest memiliki akurasi jauh lebih tinggi karena model ini sudah melalui proses pelatihan (training) dengan data yang memungkinkannya belajar pola dan hubungan fitur secara optimal. Sedangkan sistem Fuzzy Logic pada kasus ini belum dioptimalkan melalui pelatihan data yang memadai, sehingga aturan dan fungsi keanggotaannya kurang mampu menangkap kompleksitas pola pada data, sehingga performanya jauh lebih rendah.



Block Diagram:



**Thank you
very much!**