LAPORAN TUGAS 1

disusun untuk memenuhi
tugas mata kuliah Pembelajaran Mesin

oleh :
Alvia Zuhra (2208107010003)
Al-Mahfuzh Fadhlur Rohman (2208107010016)
Nurul Uzratun Nashriyyah (2208107010030)
Ganang Setyo Hadi (2208107010052)
Azimah Al-Huda (2208107010069)

JURUSAN INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS SYIAH KUALA
2025

DAFTAR ISI

A. DATA DESCRIPTION

Nama Dataset dan Sumbernya

Dataset yang digunakan adalah Heart Failure Prediction Dataset yang diperoleh dari platform Kaggle, diunggah oleh pengguna fedesoriano pada September 2021. Dataset ini merupakan kompilasi dari 5 dataset jantung berbeda yang berasal dari UCI Machine Learning Repository. Berikut adalah link Dataset: https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction

Deskripsi Singkat tentang Dataset

Dataset ini berisi data medis pasien yang dapat digunakan untuk memprediksi kemungkinan penyakit jantung. Dataset ini signifikan karena penyakit kardiovaskular merupakan penyebab kematian nomor 1 secara global, menyebabkan sekitar 17,9 juta kematian setiap tahun. Data ini dapat membantu pengembangan model untuk deteksi dini dan pengelolaan risiko kardiovaskular, terutama bagi mereka yang memiliki faktor risiko seperti hipertensi, diabetes, atau hiperlipidemia.

Jumlah Data

- Jumlah sampel: 918 observasi
- Jumlah fitur: 11 fitur prediktor (Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, ST_Slope)
- Label: 1 kolom target (HeartDisease: 1 untuk penyakit jantung, 0 untuk normal)

Format Data

Dataset tersedia dalam format CSV (Comma Separated Values), yang merupakan format standar untuk data tabular yang mudah diproses menggunakan berbagai library pemrograman seperti Pandas dalam Python.

B. DATA LOADING

1. Library yang digunakan

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
```

Kode ini menggunakan beberapa library penting untuk analisis dan visualisasi data. Pandas (import pandas as pd) dipakai untuk manipulasi data dalam bentuk DataFrame, seperti membaca file CSV dan menghitung statistik dasar. NumPy (import numpy as np) digunakan untuk perhitungan numerik, misalnya menghitung rata-rata atau mendeteksi outlier dengan kuartil. Untuk visualisasi data, Matplotlib (import matplotlib.pyplot as plt) dan Seaborn (import seaborn as sns) digunakan untuk

membuat grafik seperti boxplot dan scatter plot dengan tampilan yang informatif. Namun, ada pengulangan pada import matplotlib.pyplot as plt yang sebaiknya dihapus agar lebih rapi. Secara keseluruhan, kombinasi library ini mempermudah proses analisis data secara efisien.

2. Memuat dataset

```
df = pd.read_csv('dataset/heart.csv')
```

Kode df = pd.read_csv('dataset/heart.csv') digunakan untuk memuat dataset bernama heart.csv yang berada di folder dataset ke dalam DataFrame Pandas dan menyimpannya dalam variabel df. Fungsi pd.read_csv() membaca file CSV dan otomatis mengenali baris serta kolomnya, sehingga data bisa langsung dianalisis.

C. DATA UNDERSTANDING

1. Mengetahui Ukuran Dataset

```
df.shape
```

Kode df.shape digunakan untuk mengetahui ukuran dataset yang tersimpan dalam DataFrame df. Berdasarkan outputnya (918, 12), dimana dapat disimpulkan bahwa dataset tersebut memiliki 918 baris yang merepresentasikan jumlah data atau sampel, serta 12 kolom yang menunjukkan jumlah fitur atau atribut.

2. Menampilkan kolom

```
df.columns
```

Kode df.columns digunakan untuk menampilkan nama-nama kolom yang ada dalam DataFrame df. Berdasarkan output yang diperoleh, dataset ini memiliki 12 kolom dengan nama seperti Age, Sex, ChestPainType, RestingBP, Cholesterol, hingga HeartDisease.

3. Menampilkan informasi ringkas dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Age             918 non-null    int64  
 1   Sex             918 non-null    object 
 2   ChestPainType   918 non-null    object 
 3   RestingBP       918 non-null    int64  
 4   Cholesterol     918 non-null    int64  
 5   FastingBS       918 non-null    int64  
 6   RestingECG      918 non-null    object 
 7   MaxHR           918 non-null    int64  
 8   ExerciseAngina  918 non-null    object 
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object 
 11  HeartDisease    918 non-null    int64  
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

Menunjukkan informasi struktur DataFrame df yang memiliki 918 baris dan 12 kolom. Setiap kolom ditampilkan dengan nama, jumlah data yang tidak kosong (Non-Null Count), dan tipe datanya (Dtype). Terdapat 6 kolom bertipe int64 (angka bulat), 1 kolom bertipe float64 (angka desimal), dan 5 kolom bertipe object (biasanya data kategorikal atau teks). Tidak ada missing values karena semua kolom memiliki 918 non-null. Ukuran memori yang digunakan oleh DataFrame ini adalah 86.2 KB.

4. Statistik deskriptif dataset

```
df.describe()
```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

Fungsi df.describe() digunakan untuk menampilkan statistik deskriptif dari kolom-kolom bertipe numerik dalam DataFrame df. Fungsi ini menghasilkan informasi seperti rata-rata (mean), nilai maksimum dan minimum, standar deviasi (std), serta kuartil (25%, 50%, 75%) dari setiap kolom numerik.

5. Distribusi variabel target

```
# Count of target variable (heart disease presence)
print("\nDistribution of target variable (HeartDisease):")
display(df['HeartDisease'].value_counts())
print(f"Percentage of patients with heart disease: {df['HeartDisease'].mean()*100:.2f}%")
```

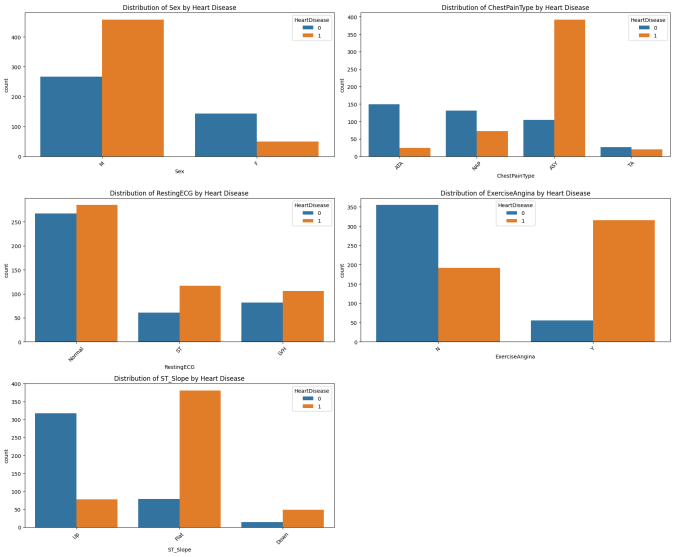
```
Distribution of target variable (HeartDisease):

HeartDisease
1    508
0    410
Name: count, dtype: int64

Percentage of patients with heart disease: 55.34%
```

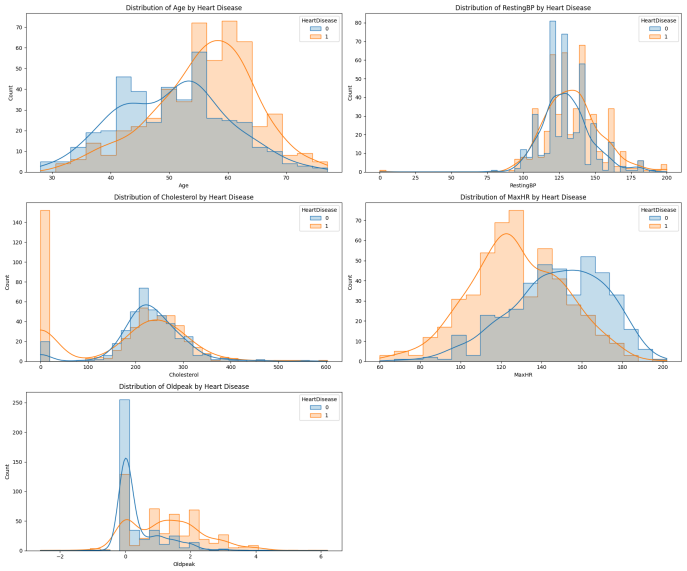
Hasil tersebut menunjukkan bahwa 55.34% pasien memiliki penyakit jantung (508 orang), sementara 44.66% tidak (410 orang), mengindikasikan ketidakseimbangan data target.

6. Menampilkan distribusi variabel kategorikal



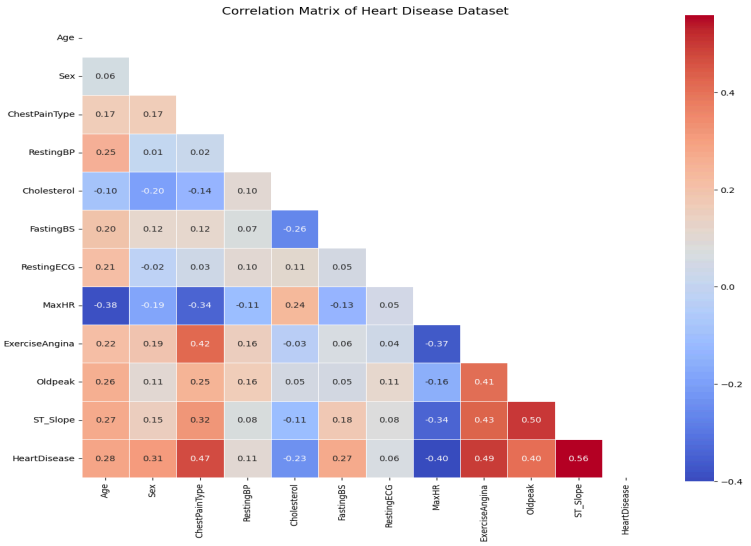
Grafik tersebut menunjukkan bahwa mayoritas penderita penyakit jantung adalah laki-laki dan tipe nyeri dada ASY menjadi indikator kuat penyakit jantung.

7. Distribusi fitur numerik



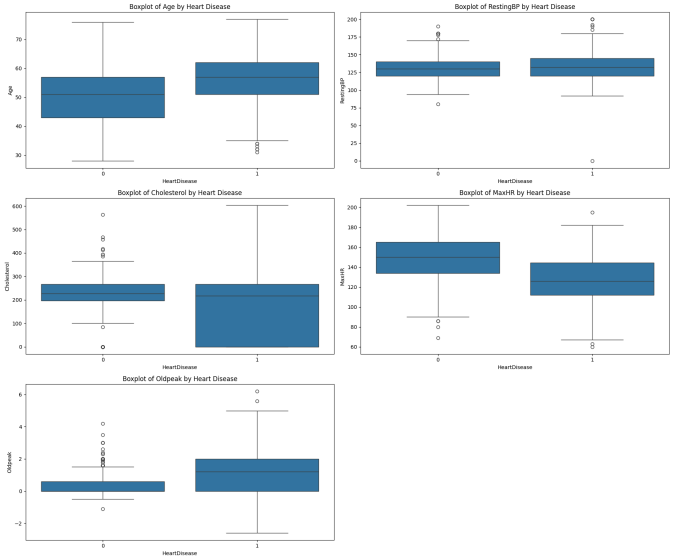
Grafik ini memperlihatkan bagaimana fitur numerik (Age, RestingBP, Cholesterol, MaxHR, dan Oldpeak) terdistribusi antara penderita penyakit jantung dan yang tidak. Penderita jantung cenderung berusia 50–65 tahun, memiliki detak jantung maksimum (MaxHR) yang lebih rendah, dan nilai Oldpeak yang lebih tinggi. Sementara itu, distribusi tekanan darah (RestingBP) hampir mirip untuk kedua kelompok. Data Cholesterol memiliki banyak nilai 0 yang tidak biasa, sehingga perlu diperiksa lebih lanjut. Pola-pola ini membantu memahami faktor risiko penyakit jantung.

8. Analisis korelasi dataset penyakit jantung



Gambar ini menunjukkan seberapa kuat hubungan antar fitur dalam dataset penyakit jantung. Angka positif berarti kedua fitur cenderung naik bersama, sedangkan angka negatif berarti saat satu fitur naik, fitur lain cenderung turun. Fitur *ST_Slope* (0.56), *Oldpeak* (0.40), dan *ChestPainType* (0.41) punya hubungan cukup kuat dengan penyakit jantung. Artinya, perubahan pada fitur-fitur ini berpengaruh besar terhadap risiko terkena penyakit jantung. Sebaliknya, *MaxHR* punya hubungan negatif (-0.40), yang berarti semakin tinggi *MaxHR*, semakin kecil kemungkinan terkena penyakit jantung. Jadi, fitur-fitur ini bisa jadi petunjuk penting dalam memprediksi penyakit jantung.

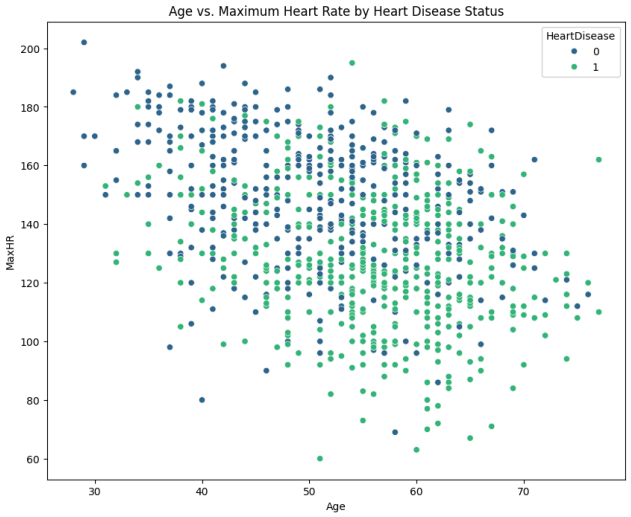
9. Boxplot fitur



Gambar ini menunjukkan perbandingan beberapa fitur seperti usia, tekanan darah, kolesterol, detak jantung maksimum (MaxHR), dan Oldpeak pada orang yang memiliki penyakit jantung (1) dan yang tidak (0). Terlihat bahwa kadar kolesterol dan Oldpeak cenderung lebih tinggi pada orang dengan penyakit jantung, sementara

MaxHR lebih rendah. Ada juga beberapa nilai yang berbeda jauh dari yang lain (outlier), terutama pada kolesterol dan tekanan darah. Secara sederhana, Oldpeak dan MaxHR tampak paling berpengaruh dalam membedakan orang yang memiliki penyakit jantung dan yang tidak.

10. Analisis Visualisasi Hubungan Antar Fitur dalam Dataset



Gambar ini menunjukkan hubungan antara usia dan detak jantung maksimum berdasarkan status penyakit jantung. Terlihat bahwa semakin tua usia, detak jantung maksimum cenderung menurun. Orang yang punya penyakit jantung (titik hijau) umumnya memiliki detak jantung maksimum yang lebih rendah dibandingkan yang sehat (titik biru).

D. DATA PREPARATION

1. Menangani nilai abnormal

- Melakukan imputasi

```
# Ganti nilai kolesterol nol dengan median terstratifikasi
zero_chol_mask = df_cleaned['Cholesterol'] == 0
zero_chol_indices = df_cleaned[zero_chol_mask].index

print(f"\nMengimputasi {len(zero_chol_indices)} nilai kolesterol...")

for idx in zero_chol_indices:
    kelompok_umur = df_cleaned.loc[idx, 'KelompokUmur']
    jenis_kelamin = df_cleaned.loc[idx, 'Sex']
    penyakit_jantung = df_cleaned.loc[idx, 'HeartDisease']

    # Dapatkan median terstratifikasi untuk grup ini
    nilai_imputasi = dapatkan_median_terstratifikasi(kelompok_umur, jenis_kelamin, penyakit_jantung)

    # Ganti nilai nol dengan median yang dihitung
    df_cleaned.loc[idx, 'Cholesterol'] = nilai_imputasi

# Hapus kolom kelompok umur sementara karena kita tidak lagi membutuhkannya
df_cleaned.drop('KelompokUmur', axis=1, inplace=True)

# Verifikasi imputasi
print(f"Nilai kolesterol nol setelah imputasi: {(df_cleaned['Cholesterol'] == 0).sum()}")
```

```
Mengimputasi 172 nilai kolesterol...
/tmp/ipykernel_10374/2698968395.py:56: Fu
  df_cleaned.loc[idx, 'Cholesterol'] = ni
Nilai kolesterol nol setelah imputasi: 0
```

Dataset ini memiliki nilai 0 pada kolesterol, yang secara medis tidak mungkin terjadi. Jika dibiarkan, nilai-nilai ini dapat menyebabkan bias dalam analisis dan model prediktif. Untuk mengatasi ini, kami menggunakan imputasi median terstratifikasi, pendekatan yang lebih cerdas dibandingkan sekadar mengganti nilai dengan median global.

Pendekatan ini lebih akurat karena:

1) Mempertahankan pola distribusi alami – Menghindari distorsi yang bisa terjadi jika mengganti semua nilai nol dengan satu angka.
2) Menghormati variasi biologis – Kolesterol bisa berbeda berdasarkan usia, jenis kelamin, dan kondisi kesehatan.
3) Lebih andal dibanding mean atau interpolasi – Mean bisa dipengaruhi oleh outlier, sedangkan interpolasi mungkin tidak mencerminkan hubungan fisiologis.

- Menangani outlier

```python
# Fungsi untuk penanganan outlier dengan metode capping berdasarkan IQR termodifikasi
def handle_outliers_with_modified_iqr(df, column, multiplier=2.5):
    """
    Menangani outlier dalam kolom numerik menggunakan pendekatan IQR termodifikasi.

    Parameters:
    df (DataFrame): DataFrame yang berisi data
    column (str): Nama kolom yang akan ditangani outliernya
    multiplier (float): Pengali untuk IQR, 2.5 lebih konservatif daripada 1.5 standar

    Returns:
    Series: Seri dengan outlier yang sudah ditangani melalui capping
    """
    # Buat salinan data untuk tidak memodifikasi data asli
    data = df[column].copy()

    # Hitung Q1, Q3, dan IQR
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1

    # Hitung batas bawah dan atas
    lower_bound = Q1 - multiplier * IQR
    upper_bound = Q3 + multiplier * IQR

    # Memastikan batas bawah tidak negatif untuk variabel yang tidak boleh negatif
    if column in ['Age', 'RestingBP', 'Cholesterol', 'MaxHR']:
        lower_bound = max(0, lower_bound)

    # Spesial untuk Oldpeak: izinkan nilai negatif karena valid secara medis

    # Lakukan capping (pembatasan)
    data_capped = data.copy()
    data_capped[data < lower_bound] = lower_bound
    data_capped[data > upper_bound] = upper_bound

    # Bandingkan sebelum dan sesudah capping
    print(f"Outlier pada kolom {column}:")
    print(f"- Jumlah sebelum capping: {sum((data < lower_bound) | (data > upper_bound))}")
    print(f"- Persentase data yang terpengaruh: {sum((data < lower_bound) | (data > upper_bound))/len(data)*100:.2f}%")
    print(f"- Batas bawah: {lower_bound:.2f}, Batas atas: {upper_bound:.2f}")

    return data_capped
```

Outlier ditangani menggunakan Capping dengan IQR Termodifikasi. Pendekatan ini menjaga preservasi informasi klinis dengan menghindari penghapusan data berharga. Dengan multiplier 2.5×IQR, metode ini lebih konservatif dibandingkan pendekatan standar (1.5×IQR), sehingga mengurangi distorsi statistik tanpa menghilangkan insight penting.

Hasil yang Didapatkan:

1) Menjaga pola distribusi alami tanpa distorsi besar
2) Mengurangi dampak outlier ekstrem tanpa menghapus informasi medis penting
3) Meningkatkan stabilitas model & kualitas prediksi

2. Encoding fitur kategorikal

```
# Binary encoding untuk Sex
df_no_outliers['Sex_Encoded'] = df_no_outliers['Sex'].map({'M': 1, 'F': 0})

# One-Hot Encoding untuk ChestPainType
chest_pain_encoded = pd.get_dummies(df_no_outliers['ChestPainType'], prefix='ChestPain')

# Ordinal Encoding untuk RestingECG (revisi)
df_no_outliers['RestingECG_Encoded'] = df_no_outliers['RestingECG'].map({'Normal': 0, 'ST': 1, 'LVH': 2})

# Binary encoding untuk ExerciseAngina
df_no_outliers['ExerciseAngina_Encoded'] = df_no_outliers['ExerciseAngina'].map({'Y': 1, 'N': 0})

# Ordinal Encoding untuk ST_Slope
df_no_outliers['ST_Slope_Encoded'] = df_no_outliers['ST_Slope'].map({'Up': 0, 'Flat': 1, 'Down': 2})

# Gabungkan hasil encoding (kecuali RestingECG yang sekarang menggunakan ordinal encoding)
df_no_outliers_encoded = pd.concat([df_no_outliers, chest_pain_encoded], axis=1)

# Hapus kolom asli jika diinginkan
columns_to_drop = ['Sex', 'ChestPainType', 'RestingECG', 'ExerciseAngina', 'ST_Slope']
df_no_outliers_encoded = df_no_outliers_encoded.drop(columns=columns_to_drop)

df_no_outliers_encoded['HeartDisease'] = df_no_outliers_encoded.pop('HeartDisease')
```

Dalam proses ini, kami menerapkan teknik encoding yang optimal untuk setiap fitur kategorikal, untuk mempertahankan esensi medis dari variabel kategorikal, efisien, informatif, dan sesuai dengan struktur data untuk analisis prediktif.

3. Fitur Scaling / Normalization

```
# Strategi scaling yang berbeda untuk fitur yang berbeda
def optimal_scaling(df):
    """
    Menerapkan metode scaling optimal untuk setiap fitur numerik
    berdasarkan karakteristik distribusinya
    """
    # Buat salinan dataframe
    scaled_df = df.copy()

    # 1. Age - StandardScaler karena distribusi mendekati normal
    age_scaler = StandardScaler()
    scaled_df['Age'] = age_scaler.fit_transform(df[['Age']])

    # 2. RestingBP - StandardScaler karena distribusi cukup simetris
    bp_scaler = StandardScaler()
    scaled_df['RestingBP'] = bp_scaler.fit_transform(df[['RestingBP']])

    # 3. Cholesterol - RobustScaler karena ada spike yang besar (outlier)
    chol_scaler = RobustScaler()
    scaled_df['Cholesterol'] = chol_scaler.fit_transform(df[['Cholesterol']])

    # 4. MaxHR - StandardScaler karena distribusi bimodal tapi cukup seimbang
    hr_scaler = StandardScaler()
    scaled_df['MaxHR'] = hr_scaler.fit_transform(df[['MaxHR']])

    # 5. Oldpeak - PowerTransformer (Yeo-Johnson) untuk mengatasi skewness
    oldpeak_scaler = PowerTransformer(method='yeo-johnson')
    scaled_df['Oldpeak'] = oldpeak_scaler.fit_transform(df[['Oldpeak']])

    return scaled_df, {
        'Age': age_scaler,
        'RestingBP': bp_scaler,
        'Cholesterol': chol_scaler,
        'MaxHR': hr_scaler,
        'Oldpeak': oldpeak_scaler
    }
```

Feature scaling adalah langkah krusial dalam pemrosesan data untuk menyamakan skala antar fitur agar model machine learning tidak terlalu condong ke fitur dengan rentang nilai yang lebih besar. Pada dataset penyakit jantung ini, kami menerapkan metode scaling yang disesuaikan dengan karakteristik distribusi setiap fitur. Tanpa scaling, fitur dengan rentang nilai besar bisa mendominasi model, sementara fitur dengan skala kecil menjadi kurang berpengaruh. Oleh karena itu, setiap fitur perlu didekati dengan metode scaling yang paling sesuai dengan distribusi datanya.

4. **Melakukan ekspor dataset yang telah diproses**

```
scaled_df.to_csv('dataset/processed/df_ready.csv', index=False)
```

Setelah melalui proses preprocessing yang matang, kini dataset telah siap digunakan dalam tahap pemodelan.