Laporan Pembelajaran Mesin

TUGAS 2
Linear dan Polynomial Regression

disusun untuk

memenuhi tugas mata kuliah

Pembelajaran Mesin

Oleh:

Kelompok VII

Nazwa Salsabila	(2108107010010)
Rizky Yusmansyah	(2208107010024)
Della Rahmatika	(2108107010041)
Zuwi Pertiwi	(2208107010061)
Berliani Utami	(2108107010082)

JURUSAN INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS SYIAH KUALA

DARUSSALAM, BANDA ACEH

2025

A. Deskripsi Dataset

Dataset yang digunakan berjudul "Personalized Medical Diet Recommendations Dataset" dari Kaggle. Dataset ini merupakan kumpulan data yang dirancang untuk memberikan rekomendasi diet yang dipersonalisasi berdasarkan informasi medis dan gaya hidup seseorang. Dataset ini terdiri dari 500 data pasien dengan total 20 kolom yang mencakup 19 fitur input dan 1 fitur target. Dataset ini bersifat tabular dan dapat digunakan untuk masalah klasifikasi, khususnya dalam menentukan jenis diet yang sesuai dengan kondisi masing-masing individu. Dataset *Personalized Medical Diet Recommendations* berisi data pasien yang mencakup informasi demografi, indikator kesehatan, gaya hidup, dan asupan nutrisi harian. Tujuannya adalah untuk memberikan rekomendasi diet yang sesuai berdasarkan kondisi personal masing-masing individu. Terdapat 19 fitur input seperti usia, jenis kelamin, BMI, tekanan darah, kolesterol, riwayat penyakit, kebiasaan olahraga, pola makan, dan lainnya, serta 1 fitur target berupa jenis diet yang direkomendasikan (seperti *Low-carb*, *Keto*, atau *Balanced*). Beberapa fitur mengandung missing values sehingga memerlukan preprocessing sebelum digunakan. Dataset ini sangat cocok untuk penelitian di bidang kesehatan, nutrisi, dan pengembangan sistem rekomendasi berbasis machine learning.

B. Data Loading

Untuk membantu dalam memahami data, ada beberapa library yang digunakan pada tahapan awal eksplorasi data yaitu mengimport pandas, numpy, matplotlib dan seaborn. Dataset kemudian dimuat ke dalam lingkungan pemrograman melalui URL dataset yang sudah di upload ke repository GitHub.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

Gambar 1 Mengimport library yg dibutuhkan

```
# URL dataset
url = "https://raw.githubusercontent.com/rizkyyus/Kelompok_7_Tugas02_Linear_Regression/refs/heads/main/Personalized_Diet_Recommendations.csv"

# Membaca dataset langsung dari URL
df = pd.read_csv(url)
```

Gambar 2 Menambahkan dataset melalui URL

C. Data Understanding

Pemahaman awal dari dataset ini adalah memahami karakteristik dasar dari dataset yang digunakan. Pada tahap ini, dilakukan eksplorasi awal terhadap dataset Personal Medical Diet guna memperoleh gambaran umum mengenai struktur data, jenis fitur, distribusi nilai, serta potensi masalah dalam data seperti nilai yang hilang, outlier, atau duplikasi.

a. Informasi Umum Dataset

Dataset ini terdiri dari 10 kolom dengan berbagai jenis data yang merepresentasikan kondisi kesehatan dan preferensi diet individu. Kolom tersebut mencakup informasi seperti Age, Gender, Height, Weight, Activity_Level, Medical_Conditions, Dietary_Preference, Daily_Calories_Intake, Nutrient_Deficiency, dan Recommended_Diet_Plan. Dataset memiliki 5000 baris, menunjukkan jumlah data yang cukup besar untuk dilakukan analisis dan pemodelan yang akurat.

```
# Menampilkan 5 baris pertama
df.head()
```

	Patient_ID	Age	Gender	Height_cm	Weight_kg	BMI	Chronic_Disease	Blood_Pressure_Systolic	Blood_Pressure_Diastolic	Cholesterol_Level	...	Protein_Intake	Carbohy
0	P00001	56	Other	163	66	24.84	NaN	175	75	219	...	105	
1	P00002	69	Female	171	114	38.99	NaN	155	72	208	...	69	
2	P00003	46	Female	172	119	40.22	NaN	137	101	171	...	183	
3	P00004	32	Female	197	118	30.41	NaN	148	91	258	...	135	
4	P00005	60	Female	156	109	44.79	Hypertension	160	109	260	...	167	

5 rows × 30 columns

```
# Menampilkan jumlah baris dan kolom
df.shape
```

```
(5000, 30)
```

Gambar 3 Informasi umum dataset

b. Informasi Struktur Dataset

```
# Menampilkan informasi tentang kolom, tipe data, dan jumlah nilai non-null
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 30 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Patient_ID                5000 non-null   object
 1   Age                       5000 non-null   int64
 2   Gender                    5000 non-null   object
 3   Height_cm                 5000 non-null   int64
 4   Weight_kg                 5000 non-null   int64
 5   BMI                       5000 non-null   float64
 6   Chronic_Disease           2957 non-null   object
 7   Blood_Pressure_Systolic   5000 non-null   int64
 8   Blood_Pressure_Diastolic  5000 non-null   int64
 9   Cholesterol_Level         5000 non-null   int64
 10  Blood_Sugar_Level         5000 non-null   int64
 11  Genetic_Risk_Factor       5000 non-null   object
 12  Allergies                 1503 non-null   object
 13  Daily_Steps               5000 non-null   int64
 14  Exercise_Frequency        5000 non-null   int64
 15  Sleep_Hours               5000 non-null   float64
 16  Alcohol_Consumption       5000 non-null   object
 17  Smoking_Habit             5000 non-null   object
 18  Dietary_Habits            5000 non-null   object
 19  Caloric_Intake            5000 non-null   int64
 20  Protein_Intake            5000 non-null   int64
 21  Carbohydrate_Intake       5000 non-null   int64
 22  Fat_Intake                5000 non-null   int64
 23  Preferred_Cuisine         5000 non-null   object
 24  Food_Aversions            3775 non-null   object
 25  Recommended_Calories      5000 non-null   int64
 26  Recommended_Protein       5000 non-null   int64
 27  Recommended_Carbs         5000 non-null   int64
 28  Recommended_Fats          5000 non-null   int64
 29  Recommended_Meal_Plan     5000 non-null   object
dtypes: float64(2), int64(17), object(11)
memory usage: 1.1+ MB
```

Gambar 4 Informasi Struktur Dataset

c. Statistik Deskriptif

Statistik deskriptif menunjukkan variasi yang cukup luas dalam variabel seperti usia, tinggi, berat badan, serta tingkat konsumsi kalori harian. Sebagian besar individu berada pada rentang usia dewasa muda hingga paruh baya, dengan kebutuhan diet yang dipengaruhi oleh tingkat aktivitas dan kondisi medis tertentu. Informasi ini penting untuk menentukan diet yang sesuai.

```
# Statistik deskriptif
print(df.describe(include="all"))
```

```
       Patient_ID          Age  Gender    Height_cm   Weight_kg          BMI  \
count        5000  5000.000000    5000  5000.000000  5000.00000  5000.000000
unique       5000          NaN       3          NaN         NaN          NaN
top         P05000          NaN  Female          NaN         NaN          NaN
freq            1          NaN    1695          NaN         NaN          NaN
mean          NaN    48.805600     NaN   174.244000    84.36620    28.353134
std           NaN    17.906991     NaN    14.229173    20.18103     8.297745
min           NaN    18.000000     NaN   150.000000    50.00000    12.630000
25%           NaN    34.000000     NaN   162.000000    67.00000    21.850000
50%           NaN    49.000000     NaN   174.000000    84.00000    27.640000
75%           NaN    64.000000     NaN   186.000000   102.00000    33.812500
max           NaN    79.000000     NaN   199.000000   119.00000    52.890000

       Chronic_Disease  Blood_Pressure_Systolic  Blood_Pressure_Diastolic  \
count             2957              5000.000000               5000.000000
unique               4                      NaN                       NaN
top           Diabetes                      NaN                       NaN
freq              1019                      NaN                       NaN
mean               NaN               133.982400                 89.735800
std                NaN                26.216215                 17.283025
min                NaN                90.000000                 60.000000
25%                NaN               111.000000                 75.000000
50%                NaN               133.000000                 90.000000
75%                NaN               157.000000                105.000000
max                NaN               179.000000                119.000000

        Cholesterol_Level  ...  Protein_Intake Carbohydrate_Intake  \
count         5000.000000  ...     5000.000000         5000.000000
unique                NaN  ...             NaN                 NaN
top                   NaN  ...             NaN                 NaN
freq                  NaN  ...             NaN                 NaN
mean           224.297800  ...      124.781800          248.590000
std             42.918923  ...       43.280037           86.535683
min            150.000000  ...       50.000000          100.000000
25%            187.000000  ...       87.000000          175.000000
50%            224.000000  ...      126.000000          249.000000
75%            261.000000  ...      162.000000          325.000000
```

Gambar 5 Statistik Deskriptif

D. Eksplorasi dan Pemrosesan Data

Tahapan pra-pemrosesan bertujuan untuk menyiapkan data mentah agar siap digunakan dalam proses analisis atau pelatihan model machine learning. Proses ini meliputi pembersihan data, transformasi, dan pembagian dataset.

a. Mengecek Missing Value

Berdasarkan hasil pengecekan terhadap dataset, diketahui bahwa terdapat missing value pada dua kolom, yaitu Chronic_Disease dan Allergies. Kolom Chronic_Disease memiliki sebanyak 2043 data yang hilang, sedangkan kolom Allergies memiliki 3497 nilai yang tidak tersedia. Hal ini menunjukkan bahwa sebagian data pada dua fitur tersebut belum terisi atau tidak tercatat. Sementara itu, seluruh kolom lainnya dalam dataset tidak mengandung missing value.

```
# Cek missing values
print("\nMissing Values:")
print(df.isnull().sum())
```

```
Missing Values:
Patient_ID                     0
Age                            0
Gender                         0
Height_cm                      0
Weight_kg                      0
BMI                            0
Chronic_Disease             2043
Blood_Pressure_Systolic        0
Blood_Pressure_Diastolic       0
Cholesterol_Level              0
Blood_Sugar_Level              0
Genetic_Risk_Factor            0
Allergies                   3497
Daily_Steps                    0
Exercise_Frequency             0
Sleep_Hours                    0
Alcohol_Consumption            0
Smoking_Habit                  0
Dietary_Habits                 0
Caloric_Intake                 0
Protein_Intake                 0
Carbohydrate_Intake            0
Fat_Intake                     0
Preferred_Cuisine              0
Food_Aversions              1225
Recommended_Calories           0
Recommended_Protein            0
Recommended_Carbs              0
Recommended_Fats               0
Recommended_Meal_Plan          0
```

Gambar 6 Mengecek missing value

b. Mengecek Jumlah Duplikasi

Mengecek jumlah duplikasi untuk menghitung jumlah baris yang duplikat dalam DataFrame. Duplikasi dapat memengaruhi kualitas analisis dan akurasi model, sehingga perlu diidentifikasi dan dibersihkan agar hasil prediksi tidak bias.

```
# Mengecek jumlah duplikasi
print(df.duplicated().sum())
```

```
0
```

Gambar 7 Penanganan missing value

c. Mengecek Outlier Bertipe Numerik

Pengecekan outlier merupakan langkah penting dalam proses pra-pemrosesan data untuk memastikan kualitas dan integritas data sebelum digunakan dalam pelatihan model. Identifikasi outlier secara dini memungkinkan pengambilan keputusan.

```
numeric_cols = df.select_dtypes(include=['int64', 'float64']).drop(columns="BMI").columns

# Deteksi outlier menggunakan IQR
def detect_outliers_iqr(data, column):
    Q1 = data[column].quantile(0.25)
    Q3 = data[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = data[(data[column] < lower_bound) | (data[column] > upper_bound)]
    return outliers

# Tampilkan jumlah outlier per kolom numerik
for col in numeric_cols:
    outliers = detect_outliers_iqr(df, col)
    print(f"{col}: {len(outliers)} outliers")
```

```
Age: 0 outliers
Height_cm: 0 outliers
Weight_kg: 0 outliers
Blood_Pressure_Systolic: 0 outliers
Blood_Pressure_Diastolic: 0 outliers
Cholesterol_Level: 0 outliers
Blood_Sugar_Level: 0 outliers
Daily_Steps: 0 outliers
Exercise_Frequency: 0 outliers
Sleep_Hours: 0 outliers
Caloric_Intake: 0 outliers
Protein_Intake: 0 outliers
Carbohydrate_Intake: 0 outliers
Fat_Intake: 0 outliers
Recommended_Calories: 0 outliers
Recommended_Protein: 0 outliers
Recommended_Carbs: 0 outliers
Recommended_Fats: 0 outliers
```

Gambar 8 Mengecek Outlier

d. Hapus Fitur yang Tidak Digunakan

Beberapa fitur dihapus tidak relevan dengan target prediksi BMI. Fitur yang dihapus yaitu :

- Kolom Kesehatan : Chronic_Disease, Blood_Pressure_Systolic, Blood_Pressure_Diastolic, Cholesterol_Level, Blood_Sugar_Level, Genetic_Risk_Factor.
- Kolom Rekomendasi Diet :Recommended_Calories, Recommended_Protein, Recommended_Carbs, Recommended_Fats, Recommended_Meal_Plan serta kolom Preferred_Cuisine, Food_Aversions, Allergies.

```python
# Hapus fitur yang tidak relevan
columns_to_drop = [
    "Chronic_Disease", "Blood_Pressure_Systolic", "Blood_Pressure_Diastolic",
    "Cholesterol_Level", "Blood_Sugar_Level", "Genetic_Risk_Factor",
    "Recommended_Calories", "Recommended_Protein", "Recommended_Carbs",
    "Recommended_Fats", "Recommended_Meal_Plan", "Preferred_Cuisine","Food_Aversions", "Allergies", "Patient_ID"
]

df_cleaned = df.drop(columns=columns_to_drop)

# Tampilkan informasi dataset setelah pembersihan
print(df_cleaned.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Age                  5000 non-null   int64
 1   Gender               5000 non-null   object
 2   Height_cm            5000 non-null   int64
 3   Weight_kg            5000 non-null   int64
 4   BMI                  5000 non-null   float64
 5   Daily_Steps          5000 non-null   int64
 6   Exercise_Frequency   5000 non-null   int64
 7   Sleep_Hours          5000 non-null   float64
 8   Alcohol_Consumption  5000 non-null   object
 9   Smoking_Habit        5000 non-null   object
 10  Dietary_Habits       5000 non-null   object
 11  Caloric_Intake       5000 non-null   int64
 12  Protein_Intake       5000 non-null   int64
 13  Carbohydrate_Intake  5000 non-null   int64
 14  Fat_Intake           5000 non-null   int64
dtypes: float64(2), int64(9), object(4)
memory usage: 586.1+ KB
None
```

Gambar 9 Hapus fitur yang tidak digunakan

e. Mengecek Kembali Missing Value

Digunakan untuk mengecek apakah masih ada nilai yang hilang (missing values) dalam setiap kolom DataFrame setelah proses pembersihan data. Hasil yang ditampilkan akan menunjukkan jumlah nilai kosong atau null di masing-masing kolom.

```python
# Mengecek kembali missing values
print(df_cleaned.isnull().sum())
```

```
Age                   0
Gender                0
Height_cm             0
Weight_kg             0
BMI                   0
Daily_Steps           0
Exercise_Frequency    0
Sleep_Hours           0
Alcohol_Consumption   0
Smoking_Habit         0
Dietary_Habits        0
Caloric_Intake        0
Protein_Intake        0
Carbohydrate_Intake   0
Fat_Intake            0
dtype: int64
```

\

Gambar 10 Encoding data kategorikal

E. Preprocessing Data

Pada preprocessing dilakukan pemilihan fitur dan penyimpanan ke dalam df_selected dilakukan untuk memastikan proses modeling hanya menggunakan data yang relevan, sehingga meningkatkan efisiensi dan akurasi model.

a. Pilih Fitur yang Akan Digunakan

Pemilihan fitur ini memilih kolom-kolom yang relevan dari data yang telah dibersihkan (df_cleaned), lalu menyimpannya dalam data frame baru bernama df_selected. Ini bertujuan agar hanya data penting yang digunakan dalam proses modeling atau analisis selanjutnya.

```
# Kolom yang akan digunakan
selected_columns = [
    "Age", "Gender", "Height_cm", "Weight_kg", "Daily_Steps",
    "Exercise_Frequency", "Sleep_Hours", "Alcohol_Consumption",
    "Smoking_Habit", "Dietary_Habits", "Caloric_Intake",
    "Protein_Intake", "Carbohydrate_Intake", "Fat_Intake", "BMI"
]

# Ambil subset dari df_cleaned
df_selected = df_cleaned[selected_columns].copy()
```

Gambar 11 Memilih fitur yang digunakan

b. Memisahkan Fitur dan Target

Digunakan untuk memisahkan fitur (variabel input) dan target (nilai yang ingin diprediksi, yaitu BMI), sehingga data siap digunakan untuk proses pelatihan model.

```
# Pisahkan fitur dan target
X = df_selected.drop(columns="BMI")
y = df_selected["BMI"]
```

Gambar 12 Memisahkan Fitur dan Target

c. Pemrosesan Fitur Kategorikal dan Numerikal

Memisahkan fitur dalam dataset menjadi dua jenis, yaitu kategorikal dan numerikal untuk membuat preprocessing pipeline, di mana fitur kategorikal dan numerik diproses dengan metode yang berbeda, lalu digabung kembali untuk digunakan dalam model machine learning.

```
# Tentukan kolom kategorikal dan numerikal
categorical_features = ["Gender", "Alcohol_Consumption", "Smoking_Habit", "Dietary_Habits"]
numeric_features = X.drop(columns=categorical_features).columns.tolist()
```

Gambar 13 Memisahkan fitur kategorikal dan numerik

d. Membuat Sebuah Pipeline

Membuat sebuah pipeline preprocessing yang otomatis menormalisasi kolom numerik dengan StandardScaler, dan mengubah kolom kategorikal menjadi angka menggunakan OneHotEncoder.

```
# Preprocessing pipeline
preprocessor = ColumnTransformer([
    ("num", StandardScaler(), numeric_features),
    ("cat", OneHotEncoder(drop="first"), categorical_features)
])
```

Gambar 14 Membuat pipeline

e. Membagi Data : Train-Test Split

Dataset dibagi menjadi 80% untuk melatih model (training) dan 20% untuk menguji performa model (testing). Pembagian ini bertujuan agar model bisa belajar dari sebagian data dan diuji pada data baru. Penggunaan random_state=42 memastikan hasil pembagian tetap konsisten setiap kali dijalankan.

```
# Split data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

print(f"Jumlah data training: {X_train.shape[0]}")
print(f"Jumlah data testing: {X_test.shape[0]}")
```

```
Jumlah data training: 4000
Jumlah data testing: 1000
```

Gambar 15 Membagi data

f. Transform Data

Preprocessing diterapkan ke data latih dengan fit_transform agar model belajar pola skala dan encoding dari data. Data uji hanya ditransformasi dengan aturan yang sama untuk menjaga konsistensi dan mencegah bias.

```
# Transform data
X_train_processed = preprocessor.fit_transform(X_train)
X_test_processed = preprocessor.transform(X_test)
```

Gambar 16 Transform data

g. Melihat Hasil Preprocessing pada DataFrame

Melihat isi DataFrame hasil preprocessing bertujuan untuk memastikan semua data sudah dalam format numerik dan siap digunakan untuk pelatihan model machine learning.

```
# Ambil nama kolom dari masing-masing transformer
num_cols = numeric_features
cat_cols = preprocessor.named_transformers_["cat"].get_feature_names_out(categorical_features)

# Gabungkan semua nama kolom
all_feature_names = list(num_cols) + list(cat_cols)

# Buat DataFrame dari hasil transformasi
X_train_df = pd.DataFrame(
    X_train_processed.toarray() if hasattr(X_train_processed, "toarray") else X_train_processed,
    columns=all_feature_names,
    index=X_train.index
)

# Tampilkan beberapa baris pertama
print(X_train_df.head())
print(X_train_df.info())
```

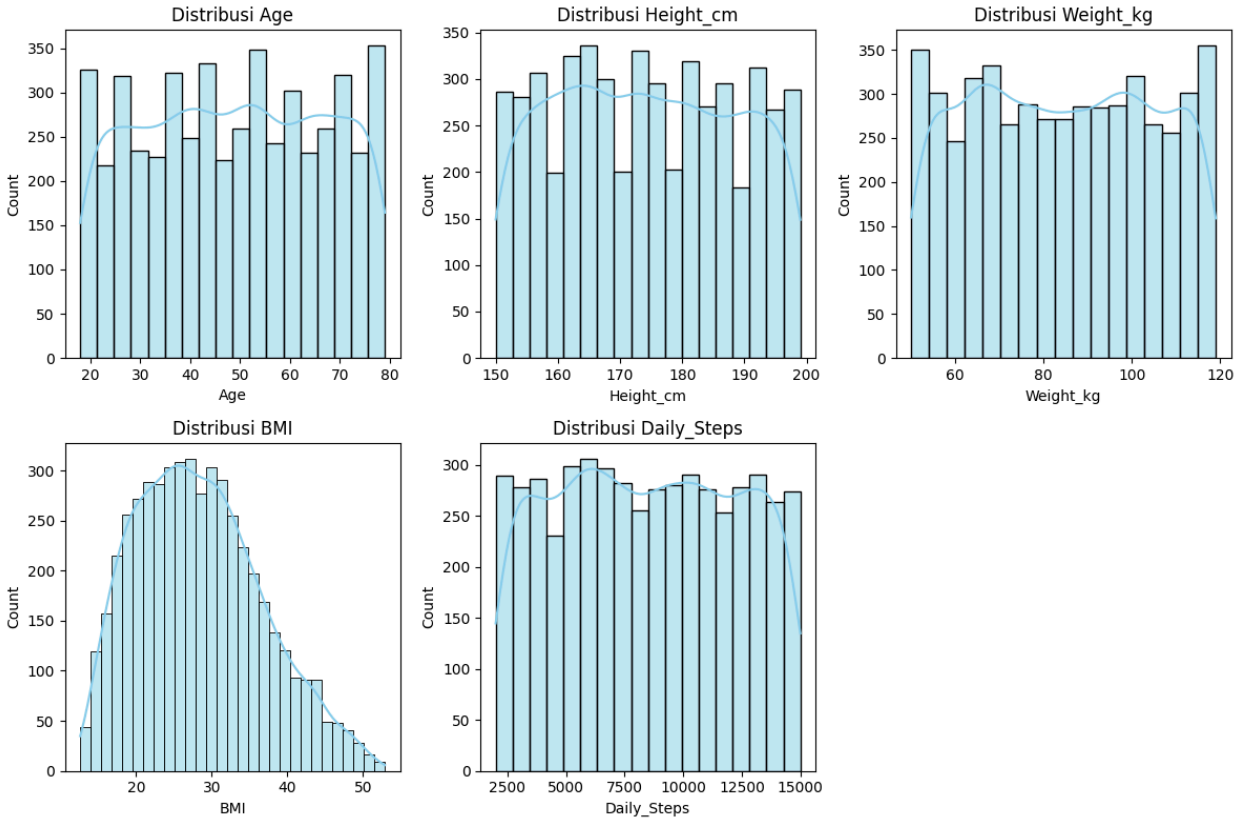
```
4227  0.350166   1.118296  -1.451792    -0.353896           -1.499909
4676 -0.095976  -1.133779   0.338763    -1.344050           -0.500969
800  -1.099797   1.751693  -0.954415    -0.280532           -1.499909
3671  0.015559   0.836787  -0.108876    -0.870105            1.496912
4193 -0.542119  -0.711515  -0.357564    -1.262711           -1.499909

      Sleep_Hours  Caloric_Intake  Protein_Intake  Carbohydrate_Intake  \
4227     0.750488       -0.322953       -0.611692            -1.127647
4676     1.042631        1.303873       -1.680139             1.250533
800      0.224629        0.297151        0.317393             0.257701
3671    -1.703519       -1.300868       -0.286512             0.707939
4193     0.984203       -1.447934       -1.680139             1.550692

      Fat_Intake  Gender_Male  Gender_Other  Alcohol_Consumption_Yes  \
4227    0.414474          1.0           0.0                      1.0
4676    0.494677          0.0           0.0                      0.0
800    -0.093476          1.0           0.0                      0.0
3671   -1.055908          0.0           0.0                      0.0
```

Gambar 17 Hasil dataframe

E. Visualisasi Distribusi Fitur

```
# --- Visualisasi Utama ---

# 1. Histogram Distribusi Fitur Numerik
# Visualisasi ini wajib untuk mengetahui sebaran data dan mendeteksi skewness
numeric_features = ['Age', 'Height_cm', 'Weight_kg', 'BMI', 'Daily_Steps']  # Changed 'height' to 'Height', 'weight' to 'Weight'
plt.figure(figsize=(12, 8))
for i, feature in enumerate(numeric_features):
    plt.subplot(2, 3, i + 1)
    # Replace data with df
    sns.histplot(df[feature].dropna(), kde=True, color='skyblue')
    plt.title(f'Distribusi {feature}')
plt.tight_layout()
plt.show()
```

Gambar 18 Visualisasi utama



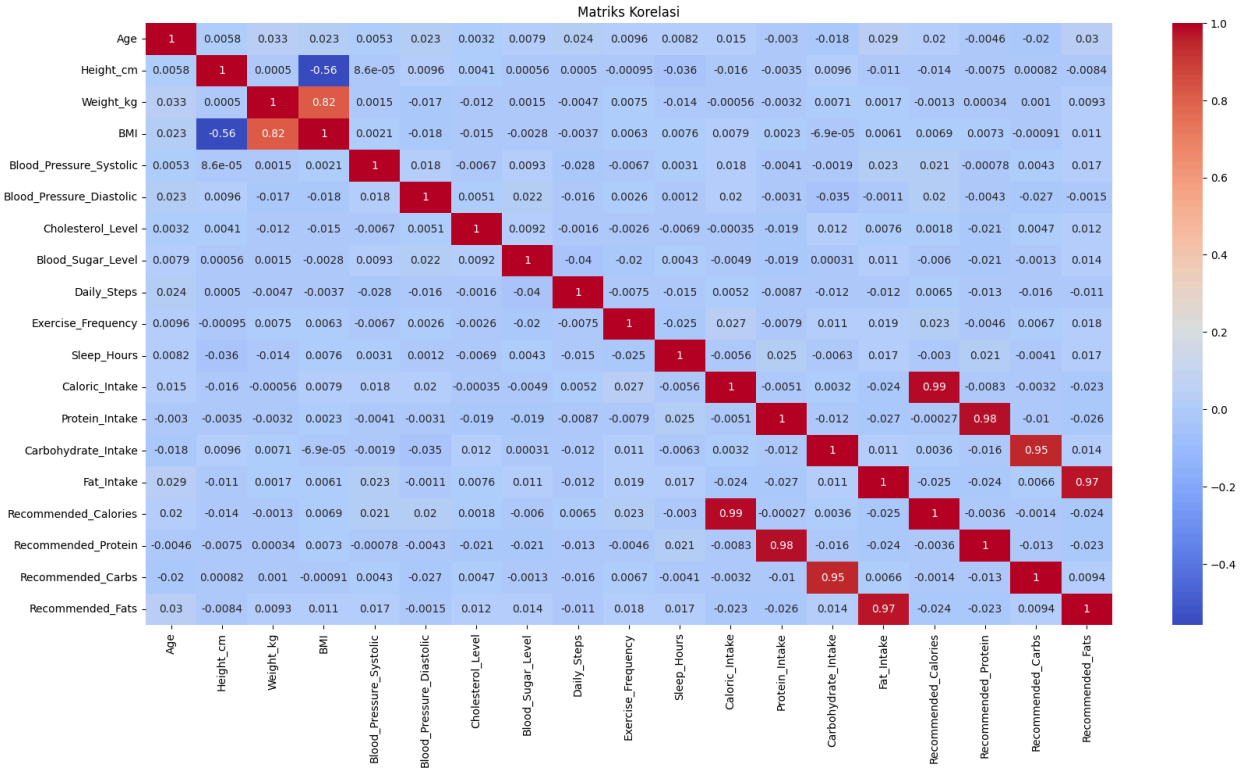
Gambar 19 Hasil visualisasi utama

Dari Visualisasi distribusi data diatas menunjukkan bahwa sebagian besar fitur dalam dataset, seperti usia, tinggi badan, berat badan, dan jumlah langkah harian, memiliki sebaran yang cukup merata tanpa adanya dominasi nilai tertentu atau outlier yang mencolok. Sementara itu, fitur BMI menunjukkan distribusi yang condong ke kanan, dengan mayoritas data berada pada kisaran

20–30, yang umumnya termasuk dalam kategori berat badan normal hingga overweight. Secara keseluruhan, distribusi data yang seimbang ini menunjukkan bahwa dataset memiliki kualitas yang baik dan siap digunakan untuk proses analisis atau pemodelan lebih lanjut.

```
plt.figure(figsize=(20,10))
# Select only numeric columns for correlation
numeric_df = df.select_dtypes(include=np.number)
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title('Matriks Korelasi')
plt.show()
```

Gambar 20 Membuat heatmap



Gambar 21 Hasil visualisasi Heatmap

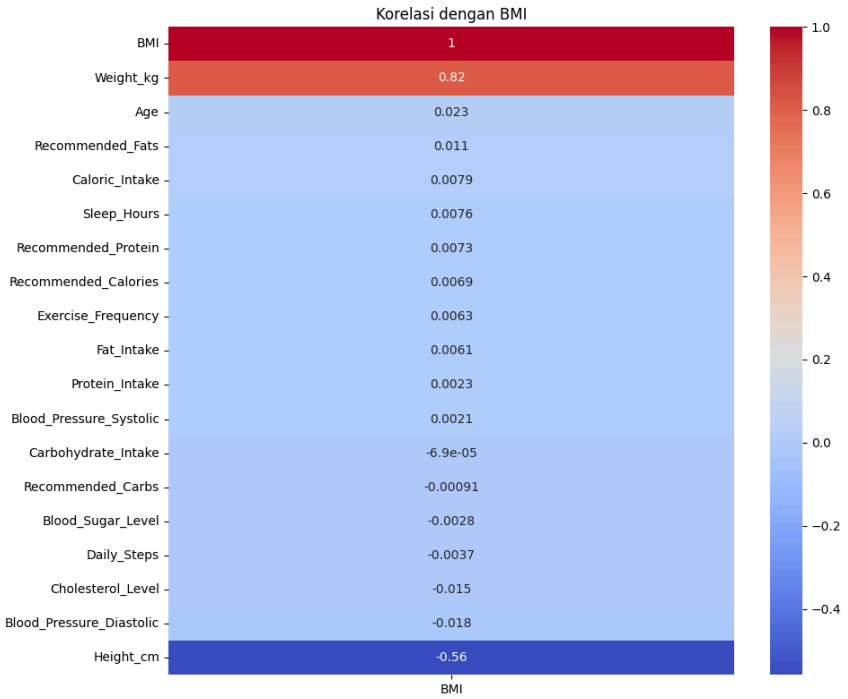
Berdasarkan hasil visualisasi heatmap korelasi diatas, dapat disimpulkan bahwa terdapat korelasi linear yang sangat kuat antara variabel konsumsi gizi (Caloric_Intake, Protein_Intake, Carbohydrate_Intake, dan Fat_Intake) dengan nilai rekomendasi masing-masing. Selain itu, berat

badan juga memiliki korelasi positif yang tinggi terhadap BMI, sedangkan tinggi badan berkorelasi negatif sedang terhadap BMI.

Sebaliknya, sebagian besar fitur lainnya seperti usia, jumlah langkah harian, kadar gula darah, dan durasi tidur menunjukkan korelasi yang sangat lemah antar variabel. Hal ini mengindikasikan bahwa hubungan antar fitur-fitur tersebut cenderung bersifat non-linear atau dipengaruhi oleh variabel lain di luar data yang tersedia. Secara keseluruhan, heatmap ini membantu dalam mengidentifikasi fitur-fitur yang redundan atau tidak relevan untuk pemodelan, serta memberikan gambaran awal tentang keterkaitan antar variabel dalam dataset.

```
# Korelasi numerik dengan target
numerik = df.select_dtypes(include=['int64', 'float64']).corr()
plt.figure(figsize=(10, 8))
sns.heatmap(numerik[['BMI']].sort_values('BMI', ascending=False), annot=True, cmap='coolwarm')
plt.title('Korelasi dengan BMI')
plt.tight_layout()
plt.show()
```

Gambar 22 Membuat korelasi BMI



Gambar 23 Visualisasi Korelasi BMI

Berdasarkan visualisasi heatmap korelasi terhadap BMI, dapat disimpulkan bahwa variabel yang paling berpengaruh terhadap BMI adalah berat badan (Weight_kg) dengan korelasi positif yang sangat kuat, yaitu sebesar 0.82. Ini menunjukkan bahwa semakin tinggi berat badan seseorang, semakin tinggi pula nilai BMI-nya. Sebaliknya, tinggi badan (Height_cm) memiliki korelasi negatif yang cukup kuat sebesar -0.56, yang berarti semakin tinggi seseorang, cenderung memiliki BMI yang lebih rendah. Sementara itu, variabel-variabel lainnya seperti usia, pola makan, jumlah langkah harian, dan tekanan darah memiliki korelasi yang sangat lemah terhadap BMI, sehingga pengaruhnya tidak signifikan dalam konteks hubungan linear. Heatmap ini membantu dalam mengidentifikasi fitur-fitur yang paling relevan untuk dianalisis atau dimasukkan dalam model prediksi BMI.

F. Implementasi Model

Tahap ini adalah tahapan dalam membangun model Linear Regression dan Polynomial Regression. Pada Polynomial Regression, akan dipilih derajat polinomial yang sesuai dengan dataset. Kedua model ini akan menggunakan teknik validasi berupa train-test split.

a. Membuat Model Linear Regression

Model regresi linear telah dibangun untuk membuat model regresi linear sederhana yang dapat memprediksi nilai BMI (Body Mass Index) berdasarkan satu atau beberapa fitur yang dipilih sebelumnya dari data, misalnya seperti berat badan atau tinggi badan. Dengan menggunakan model ini, kita ingin mengetahui seberapa kuat hubungan linier antara fitur-fitur tersebut dengan BMI.

```
# Inisialisasi dan training
lin_reg = LinearRegression()
lin_reg.fit(X_train_processed, y_train)

# Prediksi
y_pred_linear = lin_reg.predict(X_test_processed)

# Evaluasi
mse_linear = mean_squared_error(y_test, y_pred_linear)
mae_linear = mean_absolute_error(y_test, y_pred_linear)
r2_linear = r2_score(y_test, y_pred_linear)

# Tampilkan hasil evaluasi
print(f"Linear Regression - MSE: {mse_linear:.2f}, MAE: {mae_linear:.2f}, R2: {r2_linear:.2f}")
```

```
Linear Regression - MSE: 1.67, MAE: 0.98, R2: 0.98
```

Gambat 24 Membuat model linear regression

Berdasarkan hasil evaluasi, diperoleh MSE sebesar 1.67 dan MAE sebesar 0.98, yang menunjukkan bahwa rata-rata kesalahan prediksi model sangat kecil. Nilai R^2 sebesar 0.98 berarti model mampu menjelaskan 98% variasi dalam data target, yang menandakan performa model sangat baik dan prediksinya sangat mendekati nilai sebenarnya.

c. Validasi Model Linear Regression

Cross-validation dengan teknik 5-fold dilakukan untuk mengevaluasi seberapa konsisten dan andal performa model regresi linear dalam memprediksi nilai BMI. Dengan membagi data latih menjadi lima bagian dan melakukan pelatihan serta pengujian secara bergantian, metode ini memberikan gambaran yang lebih menyeluruh terhadap kemampuan model dalam menghadapi variasi data. Rata-rata hasil dari lima pengujian tersebut digunakan sebagai tolok ukur performa yang lebih stabil, sehingga dapat meminimalkan risiko overfitting atau hasil evaluasi yang bias karena pembagian data tertentu. Teknik ini menjadi langkah penting dalam memastikan model memiliki kemampuan generalisasi yang baik terhadap data baru.

```
# Pipeline linear regression (dengan preprocessing aja)
linear_pipeline = Pipeline([
    ("preprocess", preprocessor),
    ("lin_reg", LinearRegression())
])

# Cross-validation (pakai 5-fold)
cv_scores_linear = cross_val_score(linear_pipeline, X_train, y_train, cv=5, scoring='r2')

# Tampilkan hasil
print(f"Cross-validated R2 scores (Linear Regression): {cv_scores_linear}")
print(f"Mean CV R2: {cv_scores_linear.mean():.2f}")
```

```
Cross-validated R2 scores (Linear Regression): [0.97679579 0.97552707 0.97795048 0.97767251 0.9776167 ]
Mean CV R2: 0.98
```

Gambar 25 Validasi model linear regression

Model Polynomial Regression derajat 2 menunjukkan akurasi sangat tinggi dengan R^2 = 1.00, MSE = 0.02, dan MAE = 0.11, yang menandakan model mampu menangkap pola data dengan sangat baik. Namun, hasil ini juga mengindikasikan kemungkinan overfitting sehingga perlu dilakukan validasi lebih lanjut.

d. Validasi Model Polynomial Regression

Model Polynomial Regression dengan Ridge Regularization dibangun menggunakan pipeline untuk menangani preprocessing, pembuatan fitur polinomial derajat 2, dan penerapan regresi dengan penalti. Tujuannya adalah menangkap pola non-linear secara akurat sekaligus mencegah overfitting, sehingga model tetap stabil dan general pada data baru.

```
from sklearn.linear_model import Ridge
from sklearn.model_selection import cross_val_score

# Buat pipeline polynomial + Ridge regression
ridge_poly_pipeline = Pipeline([
    ("preprocess", preprocessor),
    ("poly", PolynomialFeatures(degree=2)),
    ("ridge", Ridge(alpha=1.0))  # kamu bisa coba alpha=0.1, 1, 10
])

# Fit ke training data
ridge_poly_pipeline.fit(X_train, y_train)

# Prediksi dan evaluasi
y_pred_ridge = ridge_poly_pipeline.predict(X_test)

mse_ridge = mean_squared_error(y_test, y_pred_ridge)
mae_ridge = mean_absolute_error(y_test, y_pred_ridge)
r2_ridge = r2_score(y_test, y_pred_ridge)

print(f"Polynomial Ridge Regression - MSE: {mse_ridge:.2f}, MAE: {mae_ridge:.2f}, R2: {r2_ridge:.2f}")
```

```
Polynomial Ridge Regression - MSE: 0.02, MAE: 0.11, R2: 1.00
```

Gambar 26 Membuat model polynomial regresi

Model Polynomial Regression dengan Ridge Regularization menunjukkan performa sangat baik, dengan nilai MSE sebesar 0.02, MAE sebesar 0.11, dan R^2 sebesar 1.00. Hasil ini menunjukkan bahwa model sangat akurat dalam memprediksi BMI. Penggunaan Ridge membantu menjaga keseimbangan antara akurasi dan kompleksitas model, sehingga mampu menghindari overfitting dan tetap stabil saat digunakan pada data baru.

e. Evaluasi Stabilitas Model dengan Cross-Validation pada Ridge Polynomial Regression

```
cv_scores = cross_val_score(ridge_poly_pipeline, X_train, y_train, cv=5, scoring='r2')
print(f"Cross-validated R2 scores: {cv_scores}")
print(f"Mean CV R2: {cv_scores.mean():.2f}")
```

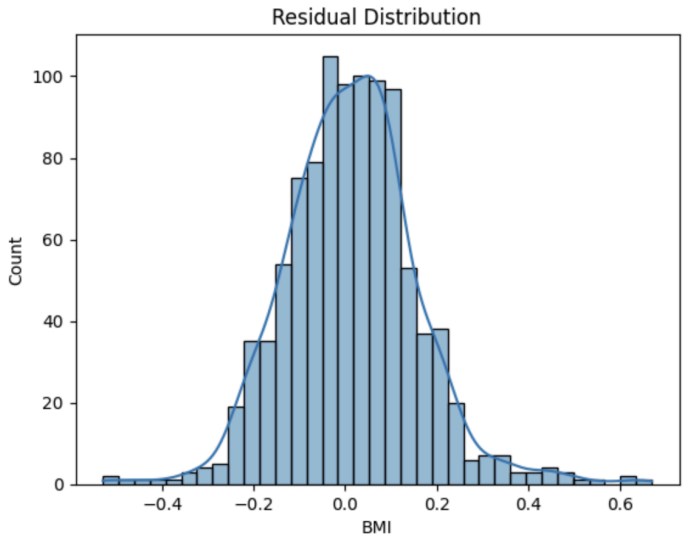
```
Cross-validated R2 scores: [0.99971411 0.99970133 0.99971998 0.99970948 0.9997053 ]
Mean CV R2: 1.00
```

Gambar 27 Model Cross-Validation

Validasi model Polynomial Ridge Regression dengan teknik 5-fold cross-validation menunjukkan hasil yang sangat baik, dengan skor R^2 pada setiap fold mendekati 1.00 dan rata-rata R^2 sebesar 1.00. Hal ini menunjukkan bahwa model tidak hanya akurat, tetapi juga konsisten dan stabil di berbagai subset data. Ridge regularization berhasil menjaga model dari overfitting sambil tetap menangkap pola non-linear dengan baik.

f. Membuat Plot Residual

```
residuals = y_test - y_pred_poly
sns.histplot(residuals, kde=True)
plt.title("Residual Distribution")
plt.show()
```



Gambar 28 Plot residual

Plot residual menunjukkan bahwa error (selisih prediksi dan nilai asli) tersebar simetris di sekitar nol dan menyerupai distribusi normal. Ini menandakan bahwa model stabil, tidak bias, dan tidak overfitting. Tidak adanya outlier besar juga menunjukkan bahwa prediksi model umumnya akurat dan dapat diandalkan.

G. Evaluasi Model

Model Linear Regression dan Polynomial Regression telah dievaluasi menggunakan metrik MSE, MAE, dan R^2 Score. Hasil evaluasi ini menunjukkan performa masing-masing model dalam memprediksi nilai Recommended_Calories, sehingga dapat dibandingkan untuk

menentukan model mana yang memberikan prediksi paling akurat dan sesuai dengan karakteristik data.

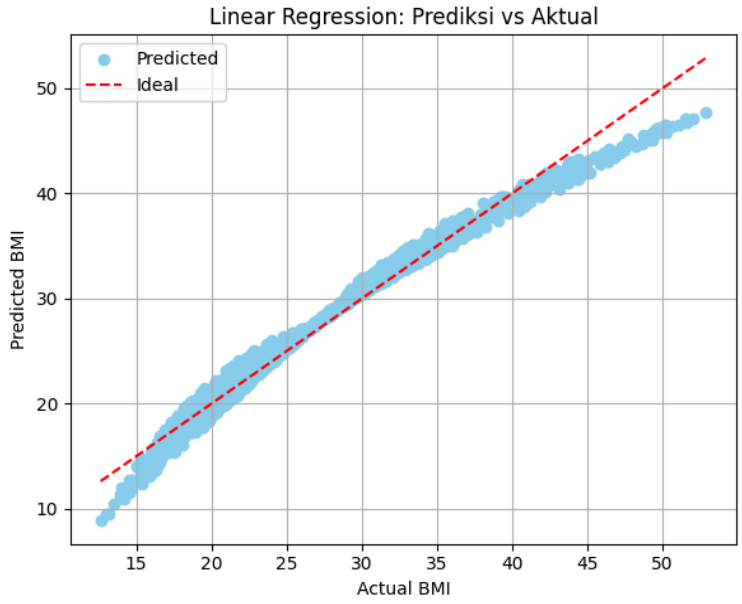
a. Evaluasi Linear Regression

Model Linear Regression telah dievaluasi menggunakan data uji dengan tiga metrik utama, yaitu Mean Squared Error (MSE), Mean Absolute Error (MAE), dan R^2 Score. Hasil evaluasi menunjukkan seberapa akurat model dalam memprediksi Recommended_Calories, serta seberapa besar variasi data target yang berhasil dijelaskan oleh fitur-fitur input.

```
# Visualisasi
plt.figure(figsize=(6, 5))
plt.scatter(y_test, y_pred_linear, color='skyblue', label='Predicted')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', label='Ideal')
plt.title("Linear Regression: Prediksi vs Aktual")
plt.xlabel("Actual BMI")
plt.ylabel("Predicted BMI")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Gambar 29 Evaluasi Linear Regressi

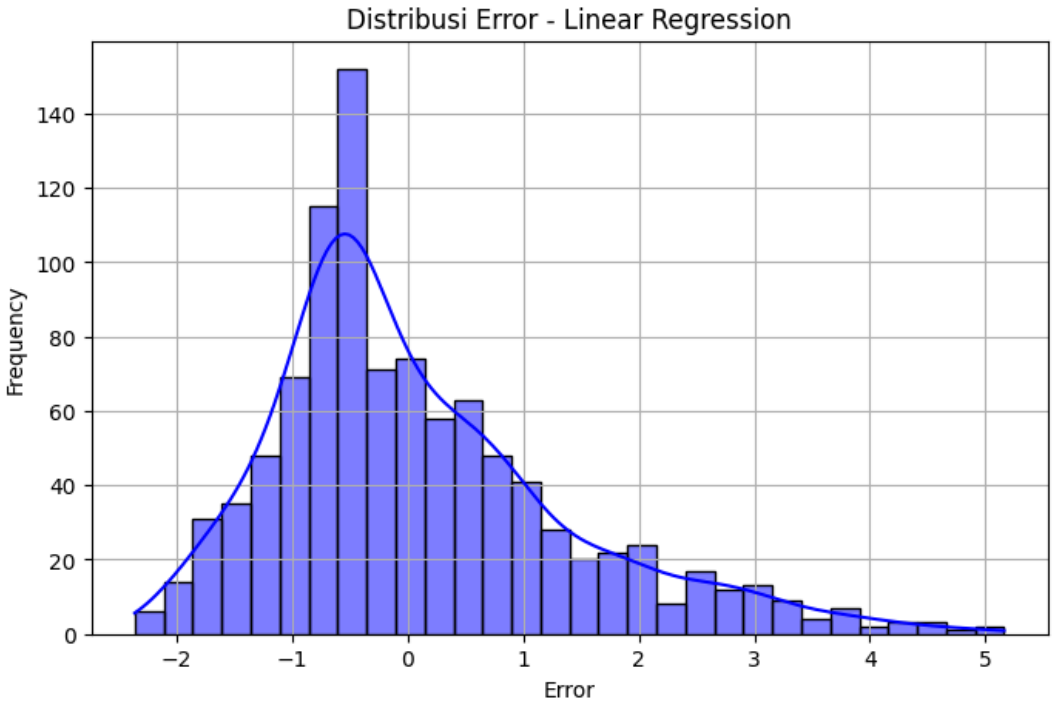
Visualisasi Linear Regressi :



Gambar 30 Visualisasi linear regressi

Berdasarkan visualisasi perbandingan antara nilai aktual dan hasil prediksi model regresi linear, dapat disimpulkan bahwa model memiliki performa yang cukup baik. Sebagian besar titik prediksi berada dekat dengan garis ideal, yang menunjukkan bahwa prediksi model secara umum mendekati nilai sebenarnya. Meskipun terdapat sedikit penyimpangan pada nilai-nilai ekstrem (sangat rendah dan sangat tinggi), tren garis menunjukkan adanya hubungan linear yang konsisten antara variabel input dan target. Dengan demikian, model regresi linear ini dinilai mampu menangkap pola data dengan akurat dan sesuai untuk digunakan dalam konteks prediksi berbasis hubungan linear.

Visualisasi Distribusi error untuk Linear regression :



Gambar 31 Visualisasi distribusi error linear regressi

Visualisasi di atas menunjukkan distribusi error dari model Linear Regression, yaitu selisih antara nilai prediksi dengan nilai aktual. Histogram yang ditampilkan memperlihatkan bahwa sebaran error tidak simetris dan cenderung condong ke kanan (positif skewed), artinya banyak prediksi yang nilainya lebih besar dari nilai aktual. Garis lengkung biru menunjukkan pola

distribusi error yang menyimpang dari distribusi normal. Hal ini menandakan bahwa model Linear Regression belum mampu menangkap pola hubungan data dengan baik. Beberapa prediksi memiliki kesalahan yang cukup besar, terutama pada sisi positif, yang menunjukkan adanya bias dalam prediksi. Dari pola distribusi ini, dapat disimpulkan bahwa model Linear Regression cenderung mengalami underfitting atau terlalu sederhana untuk menangani kompleksitas data yang sebenarnya, sehingga menghasilkan prediksi yang kurang akurat.

b. Evaluasi Polynomial Regression

Model Polynomial Regression (derajat 2) telah dievaluasi menggunakan MSE, MAE, dan R^2 Score. Hasil evaluasi menunjukkan performa model dalam menangkap hubungan non-linear antara fitur dan target. Jika nilai R^2 lebih tinggi dan MSE serta MAE lebih rendah dibanding Linear Regression, maka model polynomial lebih efektif untuk kasus ini.

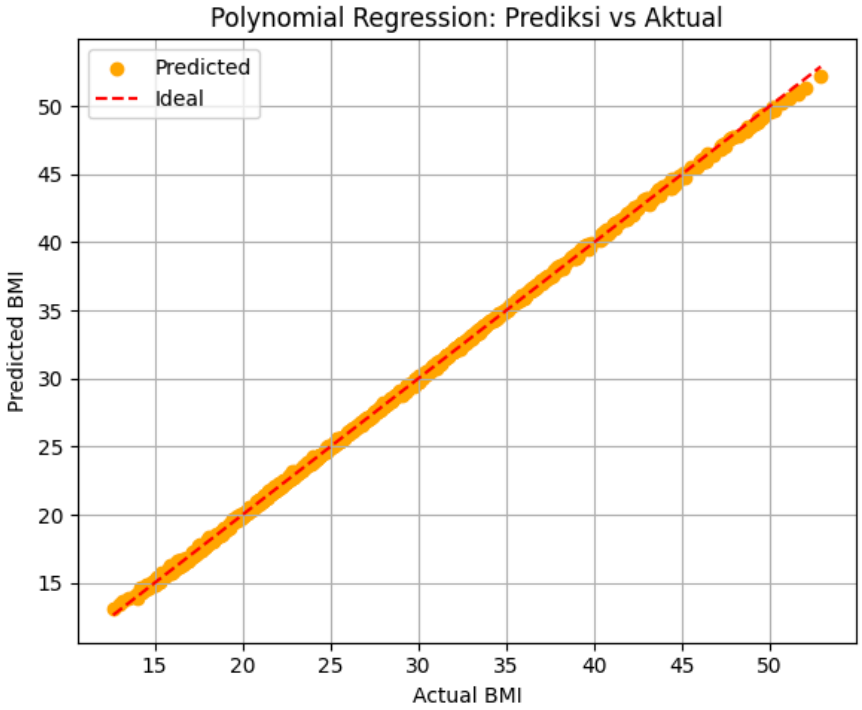
```
# Simulasi data baru dari X_test
X_new_simulated = X_test.sample(n=5, random_state=42)

# Prediksi nilai BMI dari data baru tersebut
y_pred_new = poly_pipeline.predict(X_new_simulated)

# Tampilkan hasil
hasil_df = X_new_simulated.copy()
hasil_df["Predicted_BMI"] = y_pred_new
print("\nPreview tabel prediksi:")
print(hasil_df)
```

```
Preview tabel prediksi:
      Age  Gender  Height_cm  Weight_kg  Daily_Steps  Exercise_Frequency  \
3707   72  Female        158        111         3703                   0
828    26  Female        181         63        14242                   0
2664   78    Male        157         92        13544                   1
1047   49  Female        174        102         4043                   3
3197   50    Male        169         79         3883                   4

      Sleep_Hours Alcohol_Consumption Smoking_Habit Dietary_Habits  \
3707          7.4                  No            No     Vegetarian
828           9.6                  No            No        Regular
2664          9.1                  No           Yes     Vegetarian
1047          8.2                  No            No        Regular
3197          8.6                 Yes            No           Keto

      Caloric_Intake  Protein_Intake  Carbohydrate_Intake  Fat_Intake  \
3707            3369             139                  335          69
828             2368              66                  147          53
2664            1659             164                  179          22
1047            1969              61                  319          89
3197            1544              89                  327          69
```

\

Gambar 32 Evaluasi polynomial regressi

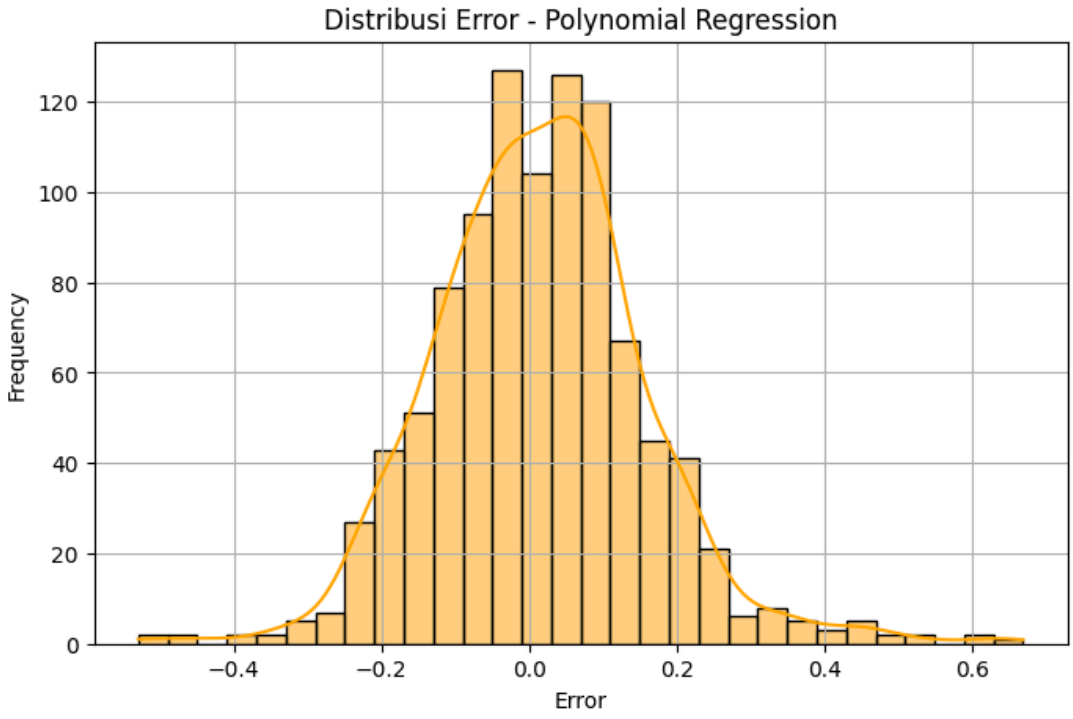
Visualisasi untuk Polynomial Regression :



Gambar 33 Visualisasi Polynomial Regressi

Visualisasi Prediksi vs Aktual pada model Polynomial Regression menunjukkan bahwa hasil prediksi BMI sangat mendekati nilai aktualnya. Hal ini terlihat dari titik-titik oranye yang sebagian besar berada tepat di atas atau sangat dekat dengan garis ideal (y = x), yang merepresentasikan prediksi sempurna. Sumbu X menunjukkan nilai BMI aktual, sedangkan sumbu Y menampilkan nilai BMI hasil prediksi model. Pola yang terbentuk menyerupai garis lurus sejajar garis ideal, menandakan bahwa model memiliki kemampuan yang sangat baik dalam mengenali pola hubungan antara fitur dan target. Tidak terdapat outlier atau penyimpangan besar yang mengindikasikan overfitting maupun underfitting. Dengan demikian, dapat disimpulkan bahwa model Polynomial Regression derajat 2 mampu memberikan prediksi yang sangat akurat, stabil, dan andal terhadap data BMI.

Visualisasi Distribusi error untuk polynomial regression :



Gambar 34 Visualisasi Error Distribusi

Berdasarkan hasil visualisasi distribusi error, dapat disimpulkan bahwa model Polynomial Regression memiliki performa yang lebih baik dibandingkan Linear Regression. Hal ini terlihat dari sebaran error pada Polynomial Regression yang lebih sempit dan terpusat di sekitar nol, menunjukkan bahwa prediksi model ini lebih akurat dan konsisten. Sebaliknya, distribusi error pada Linear Regression tampak lebih menyebar dengan variasi yang lebih besar, menandakan tingkat kesalahan prediksi yang lebih tinggi. Pola ini menunjukkan bahwa Linear Regression kurang mampu menangkap kompleksitas data, sementara Polynomial Regression lebih fleksibel dalam menyesuaikan diri terhadap hubungan non-linear, sehingga menghasilkan prediksi yang lebih presisi.

H. Analisis Hasil

Setelah dilakukan tahapan mulai dari eksplorasi data, pembersihan data, seleksi fitur, hingga pemodelan regresi, diperoleh bahwa:

1. Perbandingan antara Linear dan Polynomial Regression dalam Prediksi BMI

- Linear Regression : cukup efektif, terutama karena BMI memang secara logika dibentuk langsung dari berat dan tinggi badan. Hal ini terlihat dari kuatnya korelasi antara Weight_kg (positif) dan Height_cm (negatif) terhadap BMI. Tapi di sisi lain, fitur-fitur gaya hidup seperti Exercise_Frequency, Caloric_Intake, dan Sleep_Hours hanya menunjukkan korelasi yang sangat lemah, sehingga kurang memberikan kontribusi signifikan dalam model linier.
- Polynomial Regression : mampu menangkap pola-pola non-linear yang lebih kompleks. Hasil evaluasi memperlihatkan bahwa Polynomial Regression memiliki R^2 lebih tinggi, serta nilai error (MAE dan MSE) yang lebih rendah dibanding model linier, khususnya di data testing. Ini menunjukkan bahwa fitur-fitur tambahan mungkin memiliki pengaruh yang lebih kompleks terhadap BMI, meskipun korelasinya tidak tinggi secara linier. Distribusi error (residual) dari model Polynomial juga terlihat cukup ideal, berbentuk normal, simetris, dan terpusat di sekitar nol. Hal ini menandakan bahwa model mampu melakukan prediksi dengan stabil.

2. Model Polynomial Regression sangat sempurna sehingga perlu dicurigai terjadinya overfitting. Dari visualisasi yang ada, tidak terlihat gejala overfitting yang mencolok. Namun, karena belum ada perbandingan skor model di data training dan data testing, belum bisa disimpulkan secara penuh. Model Polynomial Regression cenderung bisa menyesuaikan data training dengan sangat baik, dikhawatirkan model kehilangan generalisasi saat digunakan untuk data baru. Sehingga, beberapa hal masih bisa dilakukan misalya dengan membuat learning curve untuk melihat performa model saat ukuran data bertambah, atau dengan menghitung metrik evaluasi yang sama pada data training untuk membandingkan performa model di data yang dikenalnya dan data baru.

3. Dari hasil korelasi, dapat dilihat bahwa BMI sangat dipengaruhi oleh Weight_kg dan Height_cm, sementara fitur gaya hidup seperti pola makan atau kebiasaan olahraga tidak punya korelasi kuat secara langsung. Namun, umumnya diketahui bahwa pola makan dan kebiasaan olahraga mempengaruhi berat badan, yang kemudian berdampak pada BMI. Jadi, ada kemungkinan adanya pengaruh tidak langsung yang tidak terdeteksi lewat korelasi Pearson biasa. Untuk langkah analisis berikutnya, bisa mencoba pendekatan bertahap dengan memodelkan berat badan berdasarkan pola makan dan aktivitas dahulu, lalu memodelkan BMI dari berat dan tinggi badan. Selain itu, juga bisa melakukan feature engineering dengan membuat rasio-rasio baru seperti perbandingan protein terhadap kalori, indeks aktivitas, atau kombinasi logika lainnya yang relevan. Terakhir, penting juga untuk menambahkan data baru dan menguji model pada data yang benar-benar belum pernah dilihat sebelumnya agar bisa mengetahui seberapa baik model bekerja secara umum.
4. Model Polynomial Regression saat ini menunjukkan performa yang unggul dan stabil untuk memprediksi BMI, karena mampu menangkap hubungan non-linear yang tidak bisa dijangkau oleh model linier. Namun, tetap perlu dilakukan analisis tambahan untuk memastikan bahwa tidak terjadi overfitting, serta eksplorasi lebih dalam terhadap fitur gaya hidup yang mungkin punya dampak tidak langsung terhadap BMI.

LINK REPOSITORY :Kelompok_7_Tugas02_Linear_Regression.ipynb - Colab