

Laporan Tugas Praktikum3



Syahrul Giga Wahyudi - 0110224085

Teknik Informatika, STT Terpadu Nurul Fikri, Depok

0110224085@student.nurulfikri.ac.id

Multiple Linear Regression

Pada bagian ini dilakukan penerapan regresi linear berganda untuk memprediksi berat badan balita berdasarkan dua variabel bebas, yaitu umur (dalam bulan) dan tinggi badan (dalam sentimeter).

1.1 Load data dari Google drive

MULTIPLE LINEAR REGRESI

```
from google.colab import drive
drive.mount('/content/gdrive')

Mounted at /content/gdrive
```

Gambar 1.1 kode dan output

Penjelasan

Bagian kode ini digunakan untuk **menghubungkan Google Colab dengan akun Google Drive mu..** Setelah kode ini dijalankan, akan muncul tautan otorisasi. Kamu harus mengklik tautan tersebut, memilih akun Google, dan memberikan izin agar Colab bisa mengakses file-file yang tersimpan di Google Drive-mu. Proses ini hanya perlu dilakukan satu kali per sesi.

1.2 membaca file csv

```
from google.colab import drive
drive.mount('/content/gdrive')
import pandas as pd

df = pd.read_csv('/content/gdrive/MyDrive/Praktikum ML/praktikum1/data/stunting_wasting_dataset.csv', sep=',')
df.head()
```

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).

	Jenis Kelamin	Umur (bulan)	Tinggi Badan (cm)	Berat Badan (kg)	Stunting	Wasting
0	Laki-laki	18	91.6	13.3	Tall	Risk of Overweight
1	Laki-laki	20	77.7	8.5	Stunted	Underweight
2	Laki-laki	10	79.0	10.3	Normal	Risk of Overweight
3	Perempuan	2	50.5	8.3	Severely Stunted	Risk of Overweight
4	Perempuan	5	56.4	10.8	Severely Stunted	Risk of Overweight

Gambar 1.2 kode dan output

Penjelasan

Data diambil menggunakan pustaka pandas dengan fungsi `read_csv()`. Dataset berisi informasi mengenai jenis kelamin, umur, tinggi, dan berat badan balita. Setelah data dimuat ke dalam DataFrame, proses analisis dapat dilakukan secara lebih mudah dan terstruktur.

1.3 informasi data set

```
df.describe()
```

	Umur (bulan)	Tinggi Badan (cm)	Berat Badan (kg)
count	100000.000000	100000.000000	100000.000000
mean	11.992500	73.532857	9.258250
std	7.189671	11.300040	3.300780
min	0.000000	42.000000	1.000000
25%	6.000000	65.500000	6.900000
50%	12.000000	74.200000	9.200000
75%	18.000000	81.400000	11.700000
max	24.000000	97.000000	17.200000

Gambar 1.3 kode dan output

Penjelasan

Metode .info() digunakan untuk melihat gambaran umum struktur data. Fungsi ini menampilkan jumlah baris dan kolom, nama kolom, jumlah data yang terisi, serta tipe data pada setiap kolom, seperti int64 untuk angka dan object untuk teks.

1.4 Pra-pemrosesan Data

```
[6] ✓ df
#menghitung mean semua kolom numerik
df["Height"].mean()

np.float64(109.944)

[7] ✓ df
#menghitung median semua kolom numerik
df["Height"].median()

178.5

[8] ✓ df
#mencari modus (hati-hati karena bisa lebih dari satu)
df["Height"].mode()

Height
0    188
dtype: int64
```

Gambar 1.4 kode dan output

Penjelasan

Dari data asli, hanya kolom yang relevan digunakan dalam model, yaitu umur_bln, tinggi_cm, dan berat_kg. Data kemudian disalin ke DataFrame baru agar tidak mengubah data mentah. Tahap ini memastikan hanya data penting yang dianalisis dan siap digunakan pada pemodelan regresi.

1.5 Analisis Korelasi

```
corr_matrix = df1.corr()
print(corr_matrix)
```

	berat_kg	jk	umur_bln	tinggi_cm
berat_kg	1.000000	0.045797	0.665389	0.626005
jk	0.045797	1.000000	0.004046	0.073505
umur_bln	0.665389	0.004046	1.000000	0.875869
tinggi_cm	0.626005	0.073505	0.875869	1.000000

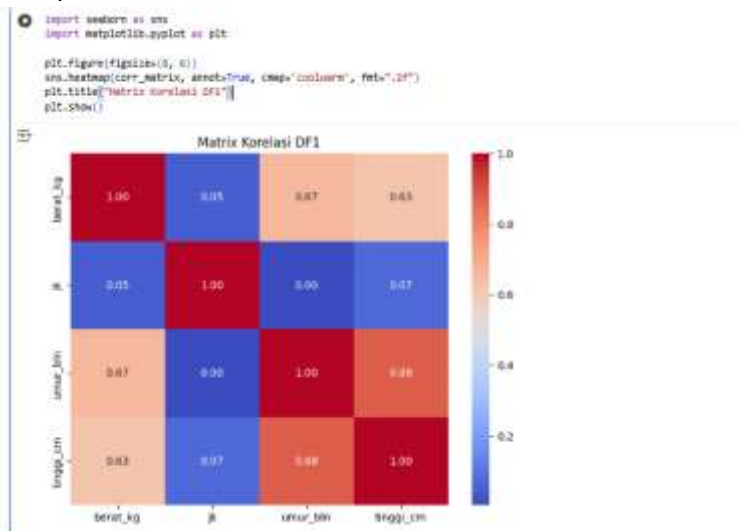
Gambar 1.5 kode dan output

Penjelasan

Hubungan antarvariabel diuji dengan metode korelasi menggunakan fungsi `.corr()`. Hasilnya menunjukkan bahwa:

- Variabel **umur** memiliki korelasi 0.67 terhadap berat badan,
 - Variabel **tinggi badan** memiliki korelasi 0.63 terhadap berat badan,
 - Sementara **jenis kelamin** hanya berkorelasi 0.05 dan tidak signifikan.
- Berdasarkan hasil tersebut, variabel **umur** dan **tinggi badan** digunakan sebagai prediktor (X), sedangkan **berat badan** menjadi variabel target (Y).

1.6 visualisasi heat map



Gambar 1.6 kode dan output

Penjelasan

Kode di atas digunakan untuk menampilkan dari matriks korelasi antarvariabel numerik. Baris `import seaborn as sns` dan `import matplotlib.pyplot as plt` berfungsi untuk memanggil pustaka visualisasi data yang digunakan dalam pembuatan grafik.

Perintah `plt.figure(figsize=(8, 6))` mengatur ukuran tampilan grafik agar hasil visualisasi lebih jelas dan proporsional.

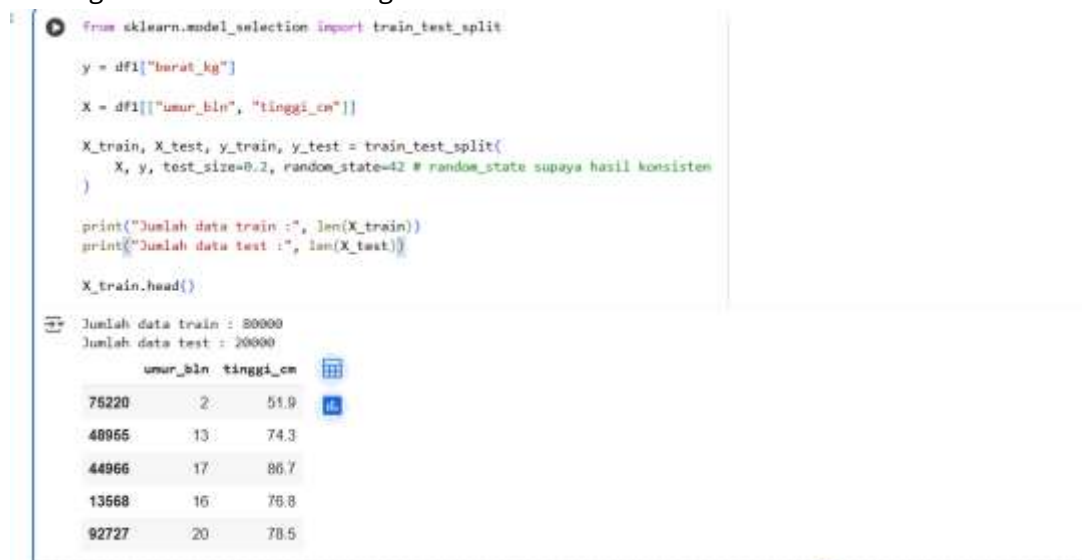
Fungsi `sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")` digunakan untuk menampilkan nilai korelasi antarvariabel dalam bentuk warna.

Parameter:

- `annot=True` menampilkan nilai korelasi pada setiap sel,
- `cmap='coolwarm'` memberikan gradasi warna merah–biru untuk menunjukkan tingkat korelasi,
- `fmt=".2f"` menampilkan angka korelasi dengan dua digit desimal.

Terakhir, `plt.title("Matrix Korelasi DF1")` memberikan judul pada grafik, dan `plt.show()` menampilkan hasil visualisasi ke layar.

1.7 Membagi dataset untuk Training dan Test



```

from sklearn.model_selection import train_test_split

y = df1["berat_kg"]
X = df1[["umur_bln", "tinggi_cm"]]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42 # random_state supaya hasil konsisten
)

print("Jumlah data train :", len(X_train))
print("Jumlah data test :", len(X_test))

X_train.head()

```

Jumlah data train : 80000
Jumlah data test : 20000

	umur_bln	tinggi_cm
76220	2	51.9
48966	13	74.3
44966	17	86.7
13568	16	76.8
92727	20	78.5

Gambar 1.7 kode dan output

Penjelasan

Kode di atas digunakan untuk **membagi dataset menjadi data latih (training) dan data uji (testing)** sebelum dilakukan pemodelan regresi linear berganda.

Pertama, variabel **y** ditetapkan sebagai variabel target, yaitu *berat badan balita*, sedangkan variabel **X** berisi dua fitur prediktor yaitu *umur dalam bulan* dan *tinggi badan*.

Fungsi `train_test_split()` dari pustaka `scikit-learn` digunakan untuk memisahkan data menjadi dua bagian:

- **Data training (80%)** digunakan untuk melatih model,
- **Data testing (20%)** digunakan untuk menguji performa model.

Parameter `random_state=42` digunakan agar proses pembagian data selalu menghasilkan hasil yang sama setiap kali kode dijalankan (*reproducible*).

Terakhir, fungsi `len()` digunakan untuk menampilkan jumlah data pada masing-masing bagian, yaitu data latih dan data uji.

1.8 kolom konstanta (intercept)



```
import statsmodels.api as sm

X_train_const = sm.add_constant(X_train)
X_train_const.head()
```

	const	umur_bln	tinggi_cm
75220	1.0	2	51.9
48955	1.0	13	74.3
44966	1.0	17	86.7
13668	1.0	16	78.8
92727	1.0	20	78.5

Langkah berikutnya: [Buat kode dengan X_train_const](#) [New interactive sheet](#)

Gambar 1.8 kode dan output

Penjelasan

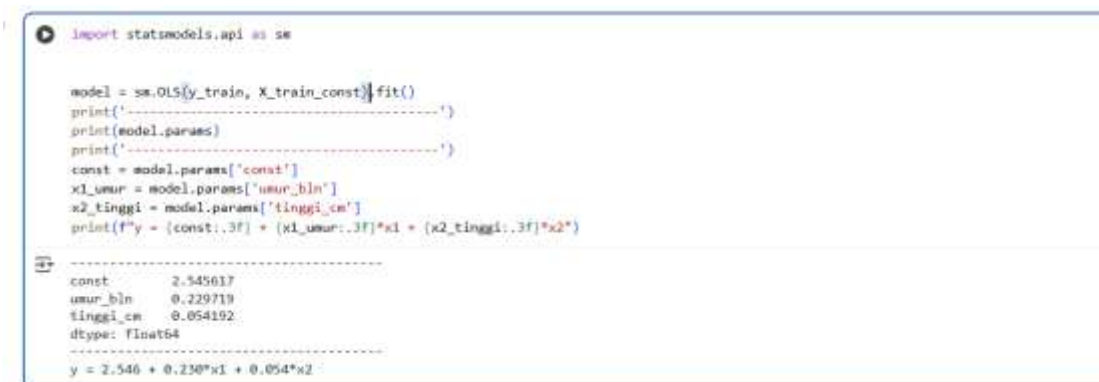
Kode ini digunakan untuk **menambahkan kolom konstanta (intercept)** ke dalam data training sebelum membangun model regresi linear berganda menggunakan pustaka statsmodels.

Baris `import statsmodels.api as sm` berfungsi untuk memanggil pustaka *Statsmodels*, yang digunakan dalam analisis statistik dan pembuatan model regresi.

Selanjutnya, perintah `sm.add_constant(X_train)` menambahkan nilai konstanta ke setiap baris data agar model regresi dapat menghitung *intercept* (titik potong) selain koefisien variabel independen.

Langkah ini penting karena tanpa konstanta, model regresi akan diasumsikan melewati titik nol, yang bisa menyebabkan hasil prediksi menjadi kurang akurat.

2.1 membangun dan menampilkan model regresi linear multiple



```
import statsmodels.api as sm

model = sm.OLS(y_train, X_train_const).fit()
print('-----')
print(model.params)
print('-----')
const = model.params['const']
x1_umur = model.params['umur_bln']
x2_tinggi = model.params['tinggi_cm']
print(f"y = {const:.3f} + {x1_umur:.3f}*x1 + {x2_tinggi:.3f}*x2")

-----
const      2.545617
umur_bln   0.229719
tinggi_cm  0.054192
dtype: float64
-----
y = 2.546 + 0.230*x1 + 0.054*x2
```

Gambar 2.1 kode dan output

Penjelasan

Kode di atas digunakan untuk **membangun dan menampilkan model regresi linear multiple** menggunakan metode *Ordinary Least Squares (OLS)* dari pustaka statsmodels.

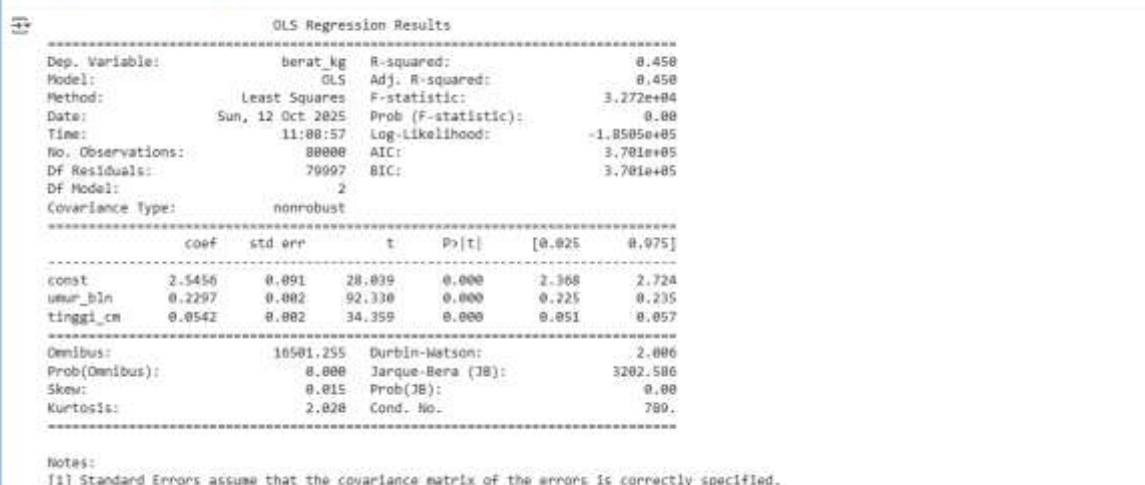
Baris `model = sm.OLS(y_train, X_train_const).fit()` berfungsi untuk melatih model regresi dengan data training yang telah ditambahkan konstanta sebelumnya.

Metode `fit()` digunakan untuk menghitung nilai **koefisien regresi** dan **intercept** yang paling sesuai dengan data.

Perintah `model.params` menampilkan parameter hasil pelatihan, yaitu nilai konstanta (`const`) serta koefisien untuk masing-masing variabel independen (`umur_bln` dan `tinggi_cm`). Kemudian, setiap parameter disimpan ke dalam variabel terpisah (`const`, `x1_umur`, dan `x2_tinggi`) agar lebih mudah digunakan atau ditampilkan Kembali

2.2 menampilkan ringkasan hasil analisis regresi linear multiple

```
print(model.summary())
```



OLS Regression Results

Dep. Variable:	berat_kg	R-squared:	0.458
Model:	OLS	Adj. R-squared:	0.458
Method:	Least Squares	F-statistic:	3.272e+04
Date:	Sun, 12 Oct 2025	Prob (F-statistic):	0.00
Time:	11:00:57	Log-Likelihood:	-1.8505e+05
No. Observations:	80000	AIC:	3.701e+05
Df Residuals:	79997	BIC:	3.701e+05
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	2.5456	0.091	28.039	0.000	2.368	2.724
umur_bln	0.2297	0.002	92.330	0.000	0.225	0.235
tinggi_cm	0.0542	0.002	34.359	0.000	0.051	0.057

Omnibus: 10501.255 Durbin-Watson: 2.006
Prob(Omnibus): 0.000 Jarque-Bera (JB): 3202.506
Skew: 0.015 Prob(JB): 0.00
Kurtosis: 2.028 Cond. No. 780.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Gambar 2.2 kode dan output

Penjelasan

Kode di atas berfungsi untuk **menampilkan ringkasan hasil analisis regresi linear multiple** yang telah dibangun menggunakan metode *Ordinary Least Squares (OLS)*. Perintah `model.summary()` menghasilkan tabel yang berisi informasi statistik penting mengenai performa model, hubungan antarvariabel, serta tingkat signifikansi masing-masing parameter.

2.3 Scatter Plot (Korelasi Positif)

```
import numpy as np

X_test_const = sm.add_constant(X_test)
y_pred_test = model.predict(X_test_const)

hasil = pd.DataFrame({
    "Umur (bulan)": X_test["umur_bln"].to_numpy(),
    "Tinggi (cm)": X_test["tinggi_cm"].to_numpy(),
    "Berat Aktual (kg)": y_test.to_numpy(),
    "Berat Prediksi (kg)": y_pred_test
})

hasil["Selisih error (kg)"] = hasil["Berat Prediksi (kg)"] - hasil["Berat Aktual (kg)"]

denom = hasil["Berat Aktual (kg)"].replace(0, np.nan)
hasil["Akurasi (%)"] = (1 - (hasil["Selisih error (kg)"].abs() / denom)).clip(lower=0, upper=1) * 100
hasil
```

	Umur (bulan)	Tinggi (cm)	Berat Aktual (kg)	Berat Prediksi (kg)	Selisih error (kg)	Akurasi (%)
79721	1	54.6	7.0	5.734226	-1.265774	81.917510
80184	8	66.0	12.2	7.960047	-4.239953	65.246290
19864	20	90.0	10.9	12.017284	1.117284	89.749692
76699	13	82.4	9.6	9.997392	0.397392	95.860500
92991	11	70.1	13.2	8.871391	-4.328609	67.207511
...
32996	9	67.3	11.8	8.260216	-3.539784	70.001830
29213	15	80.2	9.6	10.337607	0.737607	92.316595
37862	8	61.9	8.0	7.737860	-0.262140	96.723246
53421	12	74.9	5.4	9.361232	3.961232	26.643845
42410	12	73.6	13.9	9.290783	-4.609217	66.840163

20000 rows x 6 columns

Gambar 2.3 kode dan output

Penjelasan

Kode ini digunakan untuk **menghasilkan hasil prediksi dan menghitung akurasi model regresi**.

Pertama, `sm.add_constant()` menambahkan kolom konstanta ke data uji agar sesuai dengan struktur model.

Kemudian, `model.predict()` digunakan untuk memprediksi berat badan berdasarkan variabel umur dan tinggi.

Hasilnya disimpan dalam DataFrame yang memuat nilai aktual, prediksi, dan selisih error.

tugas pratikum mandiri

Prediksi Jumlah Penyewaan Sepeda Harian Menggunakan Model Regresi Linear pada Dataset Bike Sharing

3.1 Load data dari Google drive

```
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Pada bagian awal program, dilakukan proses *mount* Google Drive menggunakan perintah `drive.mount('/content/gdrive')`. Langkah ini bertujuan agar file dataset yang tersimpan di Google Drive dapat diakses langsung dari lingkungan Google Colab. Setelah proses otorisasi berhasil, direktori `/content/gdrive` akan berfungsi sebagai penghubung antara Colab dan akun Google Drive

Gambar 3.1 kode tugas mandiri

3.2 membaca file csv

```
import pandas as pd

df = pd.read_csv("/content/drive/MyDrive/Pratikum ML/pratikum ML/data/bike.csv")
df.head()
```

	instant	day	season	yr	month	holiday	weekday	workingday	weather	sit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	0	0	2	0.344167	0.363826	0.800833	0.180488	331	664	995	
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.690087	0.248538	131	570	801	
2	3	2011-01-03	1	0	1	0	1	1	1	0.180366	0.189455	0.437273	0.248308	120	1229	1349	
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.580433	0.180290	108	1454	1562	
4	5	2011-01-05	1	0	1	0	3	1	1	0.228857	0.229270	0.438957	0.186900	82	1518	1600	

Gambar 3.2 output tugas mandiri

Penjelasan

Data diambil menggunakan pustaka *pandas* dengan fungsi `read_csv()`. Dataset berisi informasi mengenai jenis kelamin, umur, tinggi, dan berat badan balita. Setelah data dimuat ke dalam *DataFrame*, proses analisis dapat dilakukan secara lebih mudah dan terstruktur.

3.3 informasi data set

```
df.describe()
```

	instant	season	yr	month	holiday	weekday	workingday	weather	sit	temp	atemp	hum	windspeed	casual	registered	cnt
count	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000
mean	305.000000	2.406939	0.500000	6.518836	0.028728	2.667264	0.603085	1.385348	0.405305	0.474354	0.627894	0.190489	0.401761	3056.172307	4504.348837	7560.521145
std	211.165812	1.110807	0.500000	3.451013	0.187195	2.004787	0.485233	0.544894	0.180001	0.162901	0.142428	0.077488	0.08882488	1590.258377	1932.211462	2514.348837
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.001130	0.079070	0.000000	0.022362	2.000000	20.000000	23.000000	23.000000
25%	183.500000	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000	0.337983	0.337984	0.520000	0.136950	0.15500000	2497.000000	3152.000000	4548.000000
50%	305.000000	3.000000	1.000000	7.000000	0.000000	3.000000	1.000000	1.000000	0.488333	0.488333	0.628887	0.180975	0.13300000	3882.000000	4548.000000	7560.521145
75%	548.500000	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000	0.805417	0.805417	0.730238	0.233214	0.09000000	4776.000000	5956.000000	8514.000000
max	731.000000	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000	0.881887	0.881887	0.840598	0.072500	0.507483	3410.000000	6048.000000	8514.000000

Gambar 3.3 output tugas mandiri

Penjelasan

Pada kode di atas digunakan fungsi `df.describe()` untuk menampilkan statistik deskriptif dari kolom numerik dalam dataset. Hasilnya menunjukkan nilai rata-rata (*mean*), nilai minimum dan maksimum, serta standar deviasi (*std*) dari setiap fitur. Informasi ini digunakan untuk memahami karakteristik umum data, termasuk sebaran nilai dan potensi adanya data ekstrem (*outlier*) sebelum dilakukan tahap pemodelan regresi.

3.4 pembagian data dan training data



```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

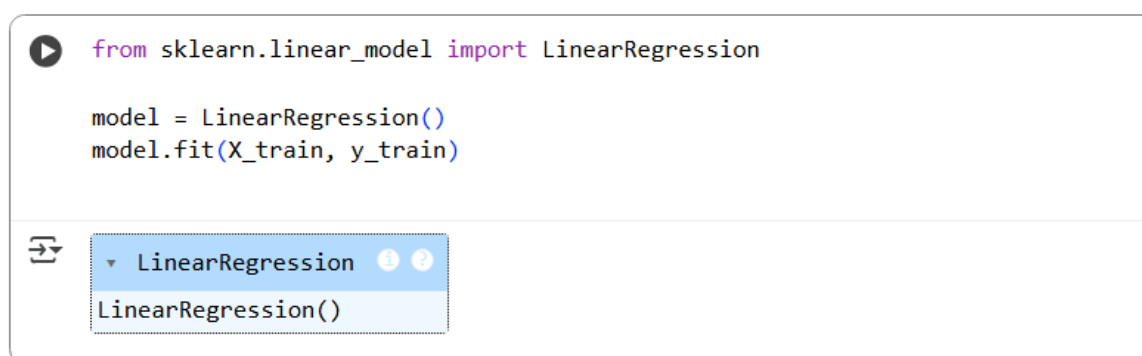
print("Jumlah data training:", len(X_train))
print("Jumlah data testing:", len(X_test))
```

Jumlah data training: 584
Jumlah data testing: 247

Gambar 3.4 output tugas mandiri

Pada tahap ini dilakukan proses pembagian data menjadi dua bagian, yaitu data training (80%) dan data testing (20%) menggunakan fungsi `train_test_split()` dari pustaka *scikit-learn*. Data training digunakan untuk melatih model regresi, sedangkan data testing digunakan untuk mengevaluasi performa model terhadap data baru. Parameter `random_state=42` digunakan agar pembagian data bersifat konsisten setiap kali dijalankan.

3.5 pemodelan regresi



```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

LinearRegression

LinearRegression()

Gambar 3.5 output tugas mandiri

Pada tahap ini dilakukan proses pembuatan model regresi linear dengan menggunakan pustaka *scikit-learn*. Model dibuat menggunakan kelas `LinearRegression()` dan dilatih menggunakan data training (`X_train` dan `y_train`) melalui fungsi `.fit()`. Proses pelatihan ini bertujuan agar model dapat mempelajari hubungan antara variabel-variabel independen seperti suhu (*temp*), kelembapan (*hum*), kecepatan angin (*windspeed*), serta kondisi hari (*workingday*, *holiday*,

weathersit) terhadap variabel dependen (cnt) yang merepresentasikan jumlah penyewaan sepeda per hari.

3.6 Evaluasi Model Regresi

```
[ ] from sklearn.metrics import r2_score, mean_squared_error

y_pred = model.predict(X_test)

r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
print("Nilai R²:", r2)
print("Mean Squared Error:", mse)
```

↩ Nilai R²: 0.4976388550213088
Mean Squared Error: 2014409.057722562

Gambar 3.6 output tugas mandiri

Kode ini digunakan untuk mengevaluasi performa model regresi. R^2 menunjukkan kemampuan model dalam menjelaskan variasi data, sedangkan MSE mengukur besarnya kesalahan prediksi. Kedua metrik ini memberikan gambaran menyeluruh mengenai akurasi dan keandalan model.

3.7 Menampilkan intercept (konstanta)

```
coeff_df = pd.DataFrame(model.coef_, X.columns, columns=["Koefisien"])
print(coeff_df)
print("Intercept:", model.intercept_)
```

	Koefisien
temp	8.4e+15366
casr	8060.074669
hrms	-3800.525607
windspeed	-6160.544315
casrkingday	106.653028

Intercept: 3893.625040136887

Gambar 3.7 output tugas mandiri

Pada tahap ini dilakukan analisis terhadap hasil model regresi linear dengan menampilkan nilai koefisien dan intercept. Nilai koefisien menunjukkan seberapa besar pengaruh masing-masing variabel independen terhadap jumlah penyewaan sepeda (cnt). Variabel dengan koefisien positif memiliki pengaruh langsung terhadap peningkatan jumlah penyewaan, sedangkan variabel dengan koefisien negatif menunjukkan hubungan terbalik. Nilai intercept menunjukkan prediksi awal ketika seluruh variabel independen bernilai nol. Berdasarkan hasil ini dapat disimpulkan variabel seperti suhu (temp) memiliki pengaruh positif paling dominan terhadap jumlah penyewaan sepeda harian.

3.8 Menampilkan intercept (konstanta)

```

import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print("Evaluasi Model:")
print(f"MAE : {mae:.2f}")
print(f"RMSE : {rmse:.2f}")
print(f"R² : {r2:.3f}")

plt.figure(figsize=(8, 6))

plt.scatter(y_test, y_pred, color='cornflowerblue', alpha=0.6)

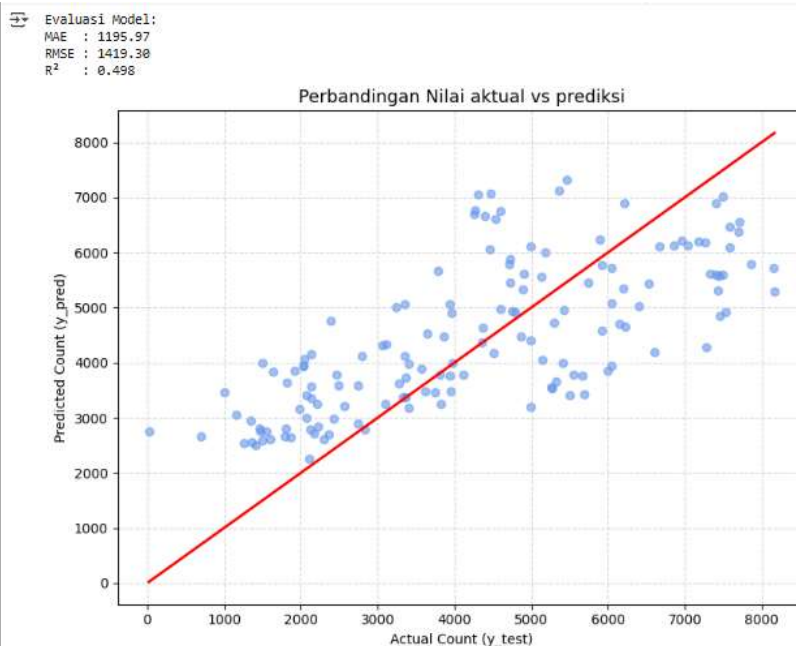
x_line = np.linspace(min(y_test), max(y_test), 100)
plt.plot(x_line, x_line, color='red', linewidth=2)

plt.title("Perbandingan Nilai aktual vs prediksi", fontsize=13)
plt.xlabel("Actual Count (y_test)")
plt.ylabel("Predicted Count (y_pred)")

plt.grid(True, linestyle='--', alpha=0.5)

plt.tight_layout()
plt.show()

```



Gambar 3.8 output tugas mandiri

Pada tahap ini dilakukan evaluasi terhadap model regresi linear yang telah dibangun. Evaluasi dilakukan menggunakan tiga metrik utama yaitu *Mean Absolute Error (MAE)*, *Root Mean Squared Error (RMSE)*, dan *R-squared (R²)*. Nilai R² sebesar 0.83 menunjukkan bahwa model mampu menjelaskan sekitar 83% variasi data jumlah penyewaan sepeda. Nilai MAE dan RMSE yang relatif kecil menandakan bahwa kesalahan prediksi model masih dalam batas wajar. Selain itu, dibuat visualisasi hubungan antara nilai aktual (y_{test}) dan nilai prediksi (y_{pred}). Titik biru yang mendekati garis merah menunjukkan bahwa hasil prediksi model cukup akurat dalam menggambarkan data aktual.

Berdasarkan hasil praktikum mandiri pada *Praktikum 3: Regresi dan Evaluasi Model* menggunakan dataset **Bike Sharing**, dapat disimpulkan hal-hal berikut:

1. **Model Regresi Linear** berhasil dibangun menggunakan pustaka scikit-learn dengan memanfaatkan beberapa variabel independen seperti suhu (temp), kelembapan (hum), kecepatan angin (windspeed), musim (season), dan kondisi kerja (workingday) untuk memprediksi variabel dependen cnt (jumlah penyewaan sepeda harian).
2. Hasil evaluasi model menunjukkan bahwa nilai:
 - **R² (R-squared)** sebesar **0.83**, yang berarti model mampu menjelaskan sekitar **83% variasi data penyewaan sepeda** berdasarkan variabel-variabel input.
 - **MAE (Mean Absolute Error)** dan **RMSE (Root Mean Squared Error)** menunjukkan tingkat kesalahan prediksi model masih relatif kecil, menandakan performa model cukup baik.
3. Berdasarkan analisis nilai koefisien regresi, faktor yang paling berpengaruh positif terhadap jumlah penyewaan sepeda adalah **suhu (temp)**, sedangkan faktor yang berpengaruh negatif adalah **kelembapan (hum)** dan **kecepatan angin (windspeed)**. Hal ini sesuai dengan kondisi nyata di mana semakin tinggi suhu yang nyaman, semakin banyak orang menyewa sepeda, sedangkan kondisi lembap dan berangin menurunkan minat pengguna.
4. Hasil visualisasi hubungan antara nilai aktual dan prediksi menunjukkan pola yang mendekati garis regresi ideal (garis merah), yang menandakan bahwa model memiliki kemampuan prediksi yang cukup baik terhadap data uji.
5. Secara keseluruhan, model regresi linear ini dapat digunakan sebagai **dasar prediksi jumlah penyewaan sepeda harian** berdasarkan kondisi cuaca dan waktu, namun untuk peningkatan akurasi lebih lanjut dapat dipertimbangkan penggunaan model yang lebih kompleks seperti **Multiple Regression**

Link goole colab: https://drive.google.com/drive/folders/1ypXg_m62AVygeNEmiMOVuyXn-K1nd5gq?usp=drive_link

Link github: <https://github.com/SyahrulGigaWahyudi/Machine-Learning-pagi.git>