ibimbing

Digital Skill Fair 35.0 Data Science

"LINNERUD DATASET MACHINE LEARNING PROJECT USING LINEAR REGRESSION"

DEWI YULIANA

ABOUT ME

"I am a dedicated physics student at the State Islamic University of Sunan Gunung Djati Bandung, specializing in robotic instrumentation physics and computational methods. With a solid foundation in Python, SQL data analysis, and artificial intelligence, I am passionate about bridging the gap between theoretical concepts and real-world applications. I have hands-on experience in AI, machine learning, and Python programming, and I'm continuously seeking opportunities to expand my knowledge in data science. "

ibimbing

TOOLS USED

ABOUT THE PROJECT

This project focuses on analyzing the Linnerud dataset using linear regression, a fundamental machine learning technique. The Linnerud dataset consists of physical exercise data, including features such as chin-ups, sit-ups, and jumps, along with health-related targets like weight, waist circumference, and pulse. The goal is to apply linear regression to predict health metrics based on exercise data and evaluate the model's performance.

TABLE OF CONTENT

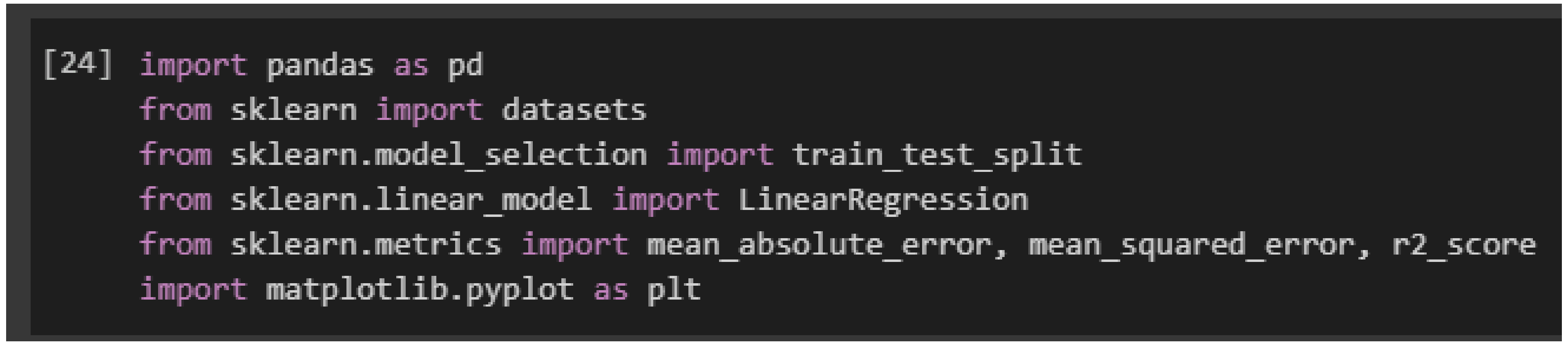
01 *Library and Dataset*

02 *Exploratory Data*

03 *Model*

04 *Predict & Evaluate*

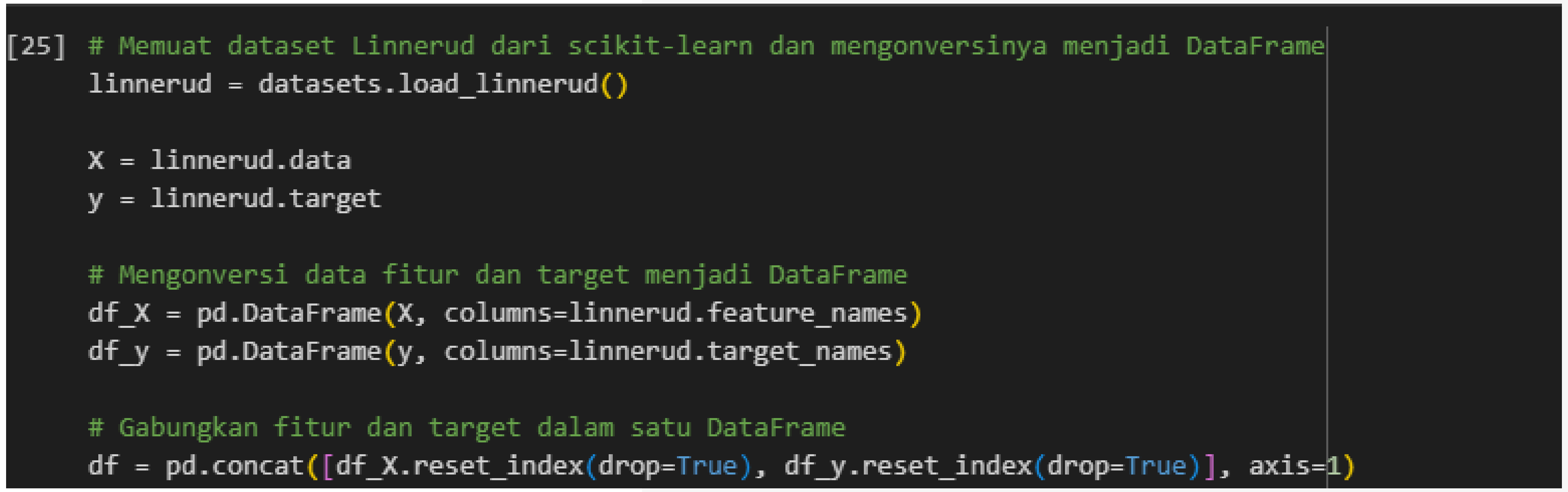
LIBRARY AND DATASET



```
[24] import pandas as pd
     from sklearn import datasets
     from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LinearRegression
     from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
     import matplotlib.pyplot as plt
```

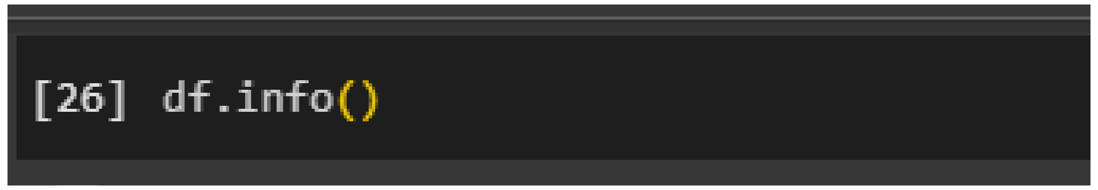
- pandas: Used for data manipulation in DataFrame format.
- numpy: Used for numerical operations and array handling.
- sklearn.datasets: For loading predefined datasets provided by scikit-learn.
- sklearn.model_selection.train_test_split: To split data into training and testing sets.
- sklearn.linear_model.LinearRegression: To create and train a linear regression model.
- sklearn.metrics: Used for evaluating the model (Mean Absolute Error, Mean Squared Error, R2 Score).
- matplotlib.pyplot and seaborn: Used for data visualization (heatmaps and scatter plots).

- The Linnerud dataset is loaded, which contains physical exercises data used to predict certain health metrics.
- X: The input features (e.g., exercises like sit-ups, cycling, etc.).
- y: The target outputs (e.g., variables like body weight, number of sit-ups, etc.).
- The data is then converted into a DataFrame for easier manipulation.

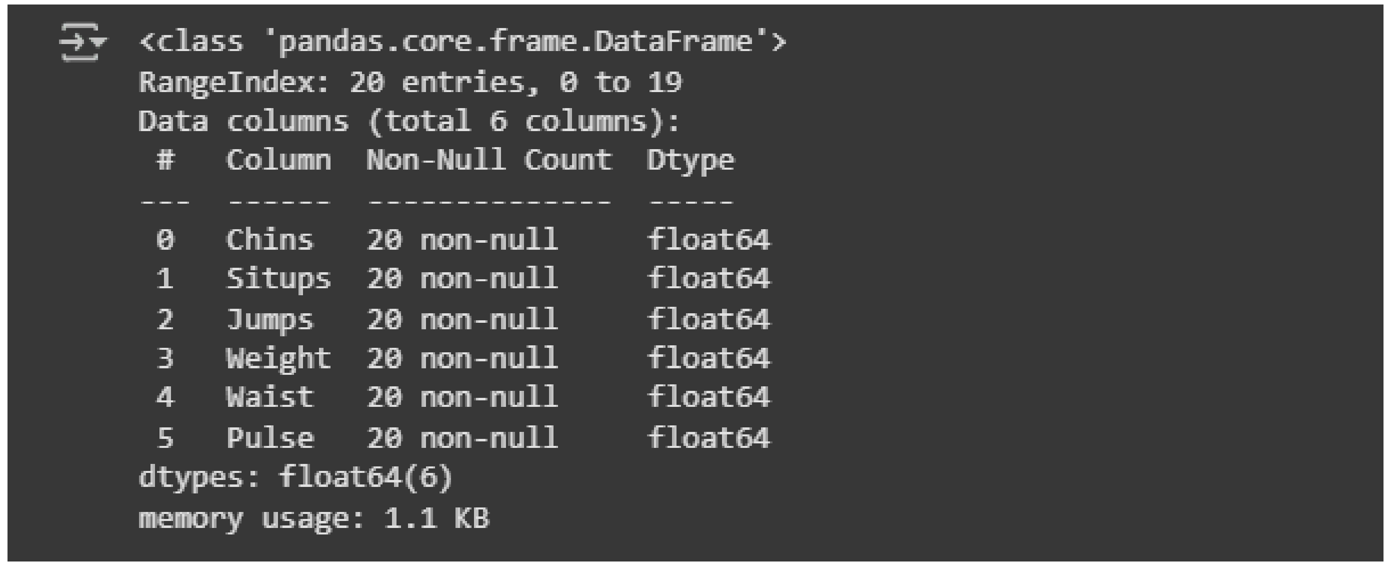


```
[25] # Memuat dataset Linnerud dari scikit-learn dan mengonversinya menjadi DataFrame
     linnerud = datasets.load_linnerud()

     X = linnerud.data
     y = linnerud.target

     # Mengonversi data fitur dan target menjadi DataFrame
     df_X = pd.DataFrame(X, columns=linnerud.feature_names)
     df_y = pd.DataFrame(y, columns=linnerud.target_names)

     # Gabungkan fitur dan target dalam satu DataFrame
     df = pd.concat([df_X.reset_index(drop=True), df_y.reset_index(drop=True)], axis=1)
```

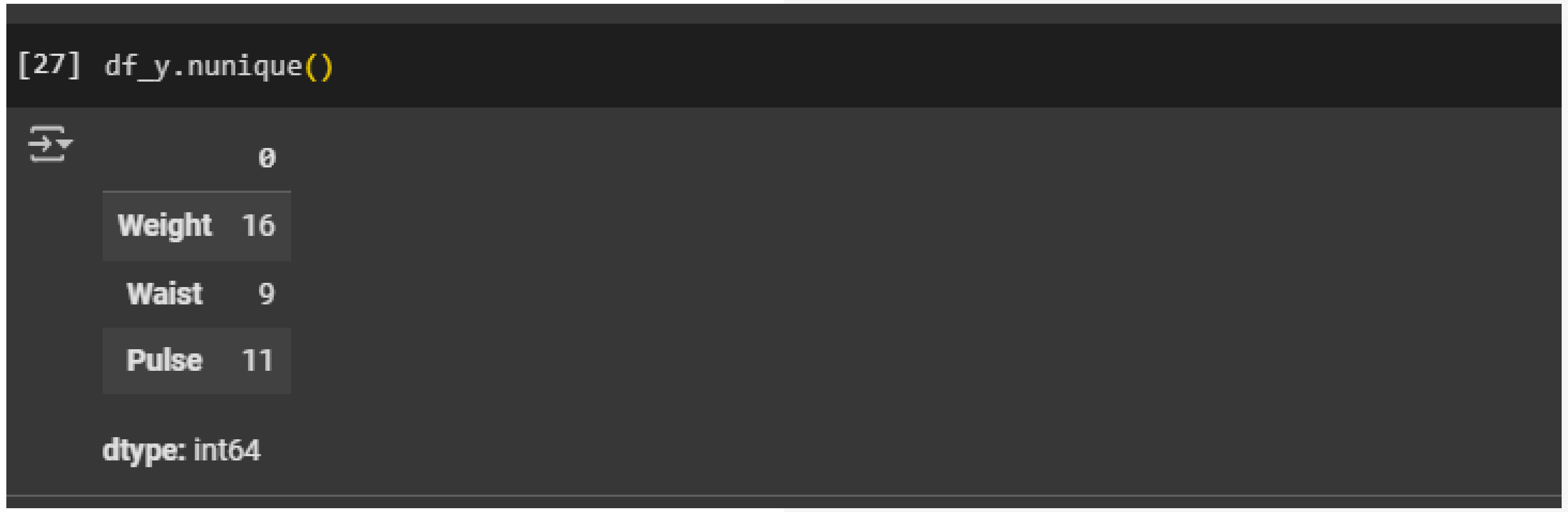
EXPLORATORY DATA



```
[26] df.info()
```



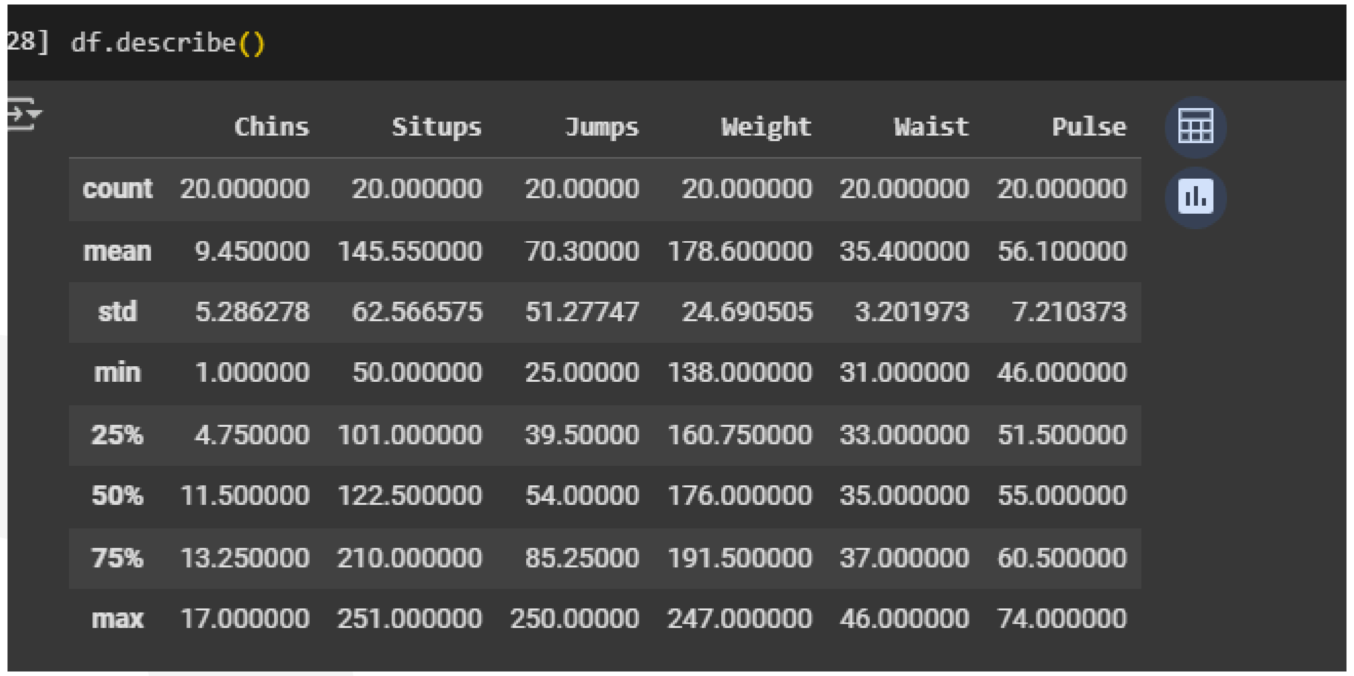
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Chins   20 non-null     float64
 1   Situps  20 non-null     float64
 2   Jumps   20 non-null     float64
 3   Weight  20 non-null     float64
 4   Waist   20 non-null     float64
 5   Pulse   20 non-null     float64
dtypes: float64(6)
memory usage: 1.1 KB
```



```
[27] df_y.nunique()
```

	0
Weight	16
Waist	9
Pulse	11

dtype: int64

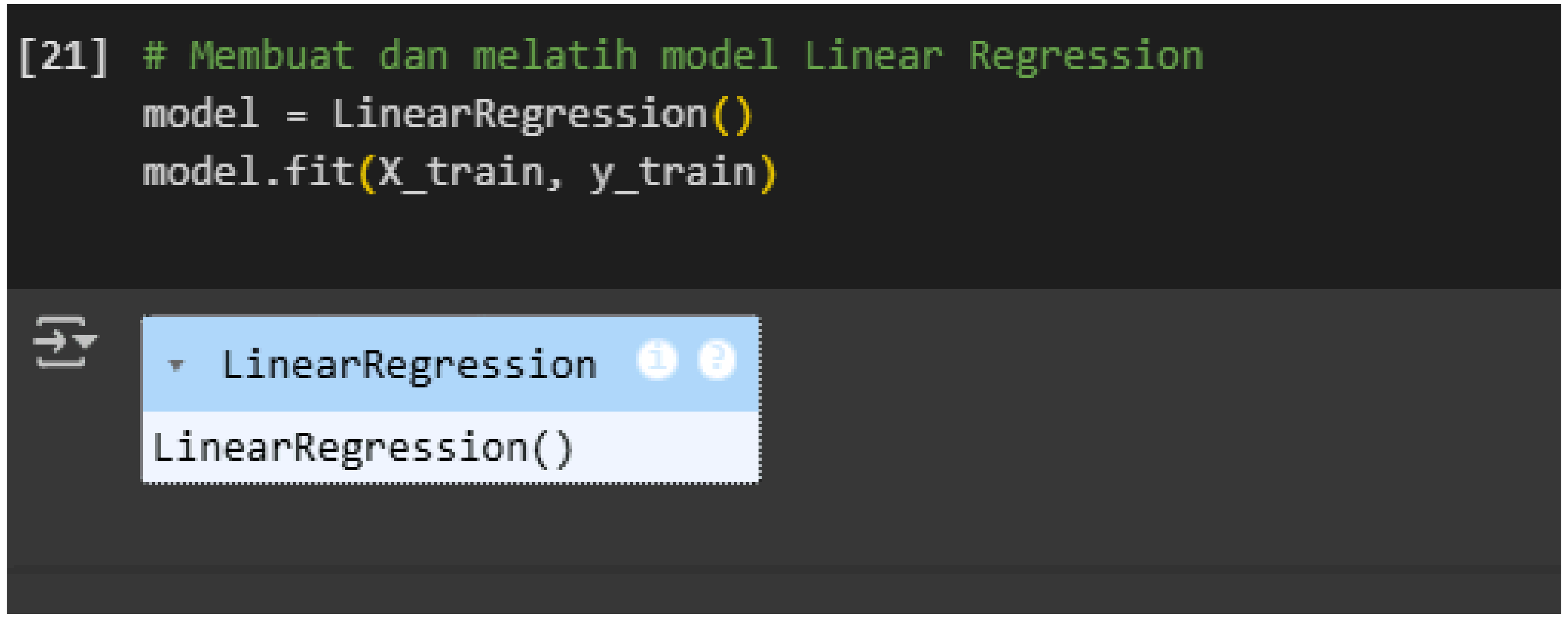


```
28] df.describe()
```

	Chins	Situps	Jumps	Weight	Waist	Pulse
count	20.000000	20.000000	20.00000	20.000000	20.000000	20.000000
mean	9.450000	145.550000	70.30000	178.600000	35.400000	56.100000
std	5.286278	62.566575	51.27747	24.690505	3.201973	7.210373
min	1.000000	50.000000	25.00000	138.000000	31.000000	46.000000
25%	4.750000	101.000000	39.50000	160.750000	33.000000	51.500000
50%	11.500000	122.500000	54.00000	176.000000	35.000000	55.000000
75%	13.250000	210.000000	85.25000	191.500000	37.000000	60.500000
max	17.000000	251.000000	250.00000	247.000000	46.000000	74.000000

- d.info() provides details about the dataset, including data types and the number of non-null entries.
- .describe() provides statistical summaries (mean, standard deviation, min, max, etc.) for each column.

MODEL



```
[21] # Membuat dan melatih model Linear Regression
     model = LinearRegression()
     model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

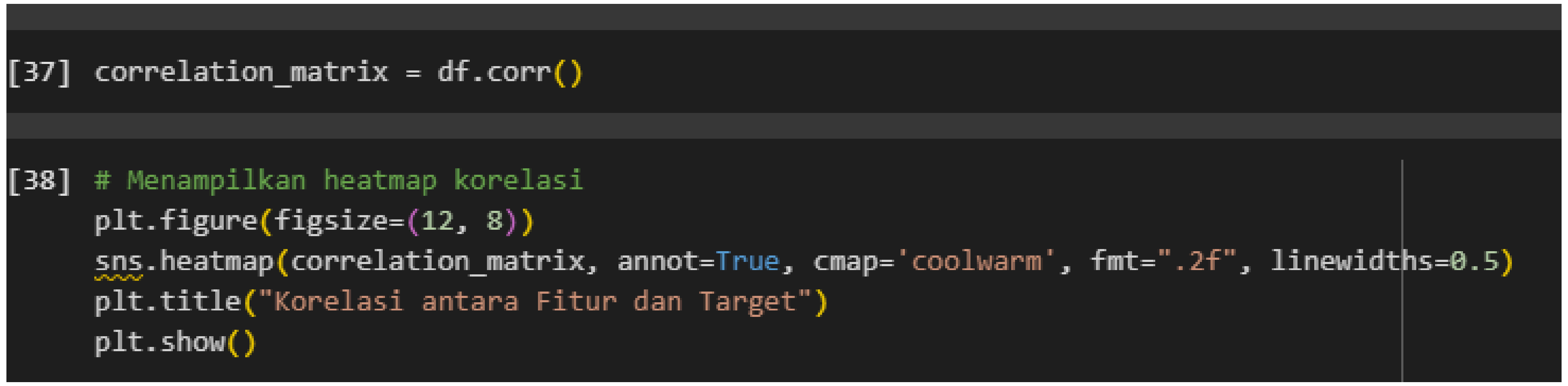
- A LinearRegression model is created.
- The .fit() method trains the model using the training data (X_train, y_train).

Linear Regression

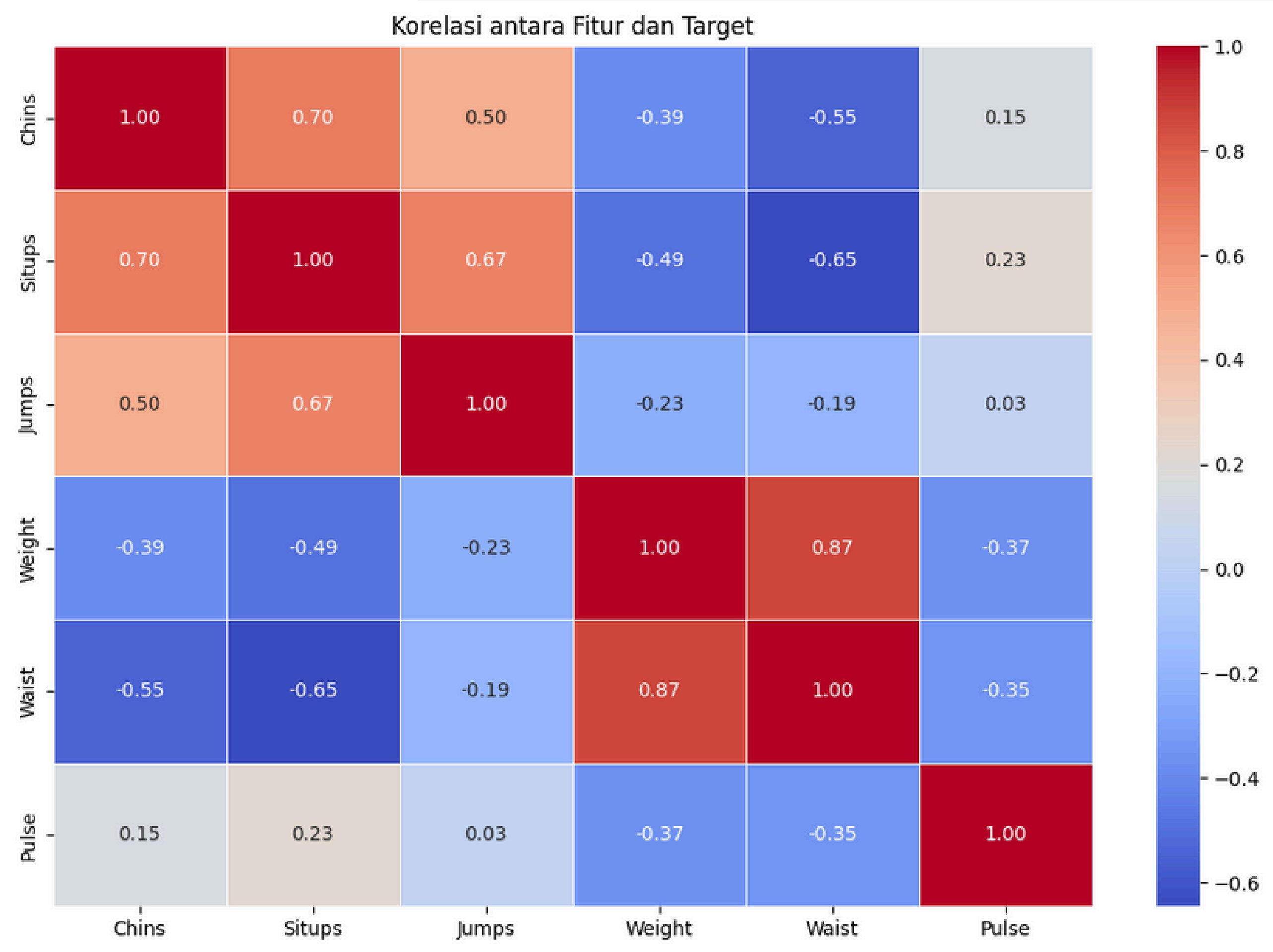
Linear Regression is a simple yet powerful algorithm used for predicting a continuous target variable based on one or more input features. In the context of the project above, the model is applied to predict multiple target variables from the Linnerud dataset using a linear relationship between the features and the targets.

PREDICT AND EVALUATE

Correlation Heatmap



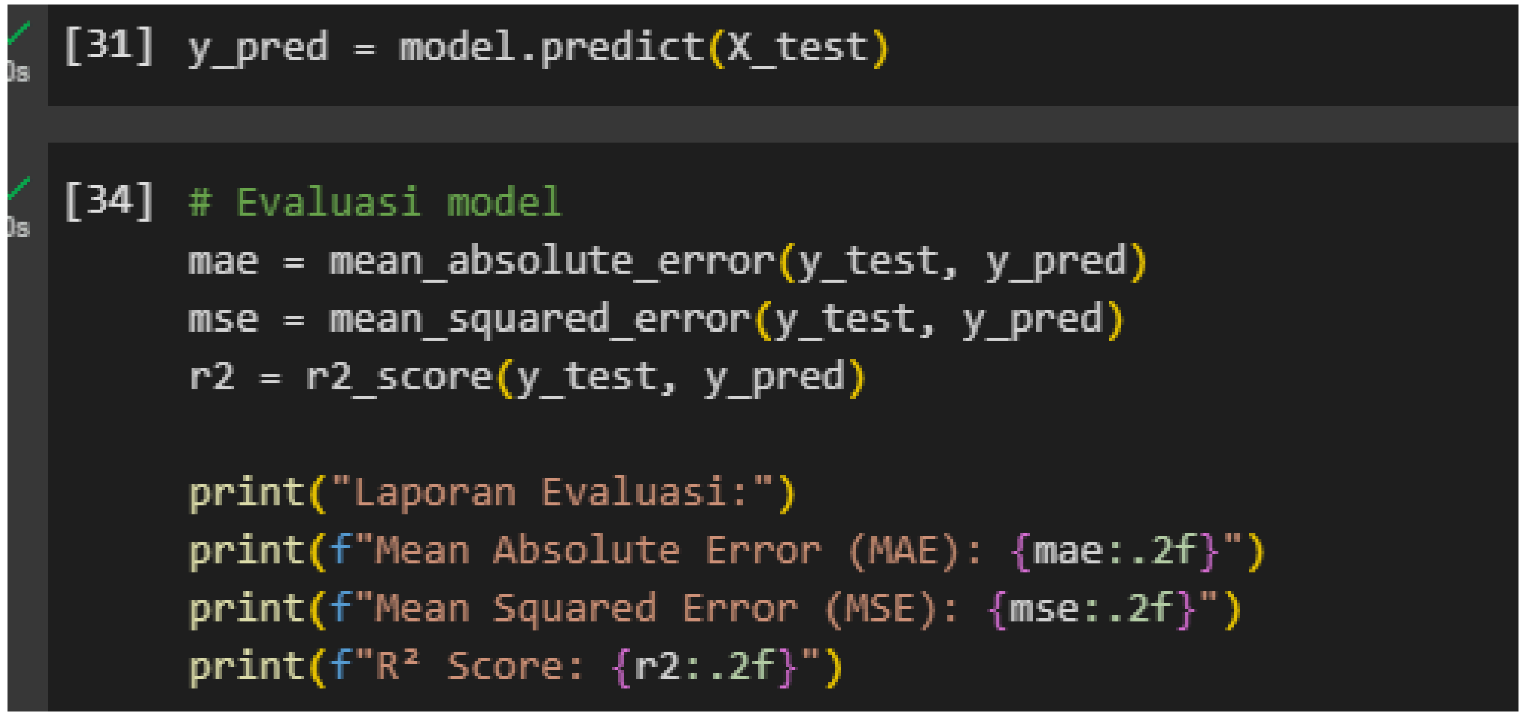
```
[37] correlation_matrix = df.corr()

[38] # Menampilkan heatmap korelasi
     plt.figure(figsize=(12, 8))
     sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
     plt.title("Korelasi antara Fitur dan Target")
     plt.show()
```



- The .corr() function calculates the correlation matrix between features and targets.
- sns.heatmap() visualizes the correlation matrix, where the color intensity represents the strength of the correlation.

PREDICT AND EVALUATE

Model Evaluation

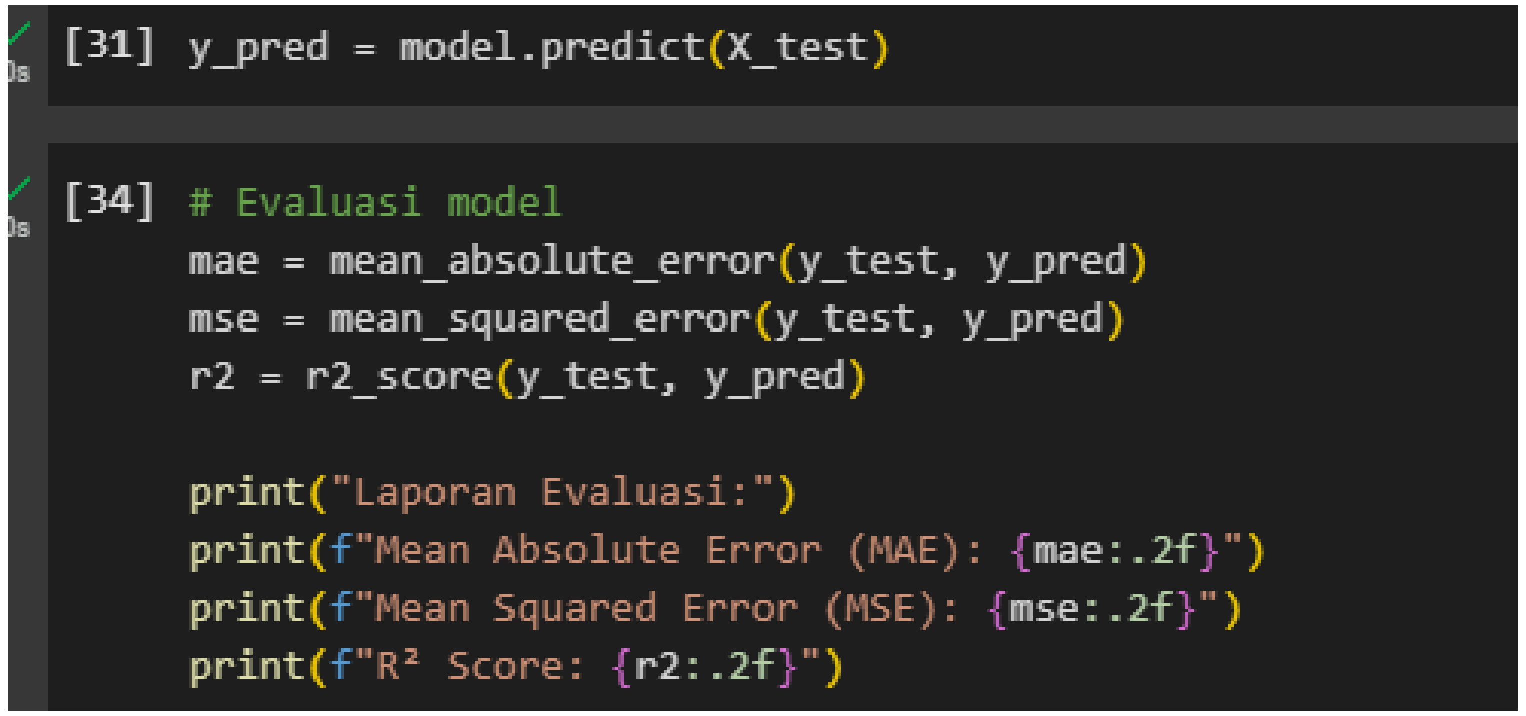


```
[31] y_pred = model.predict(X_test)

[34] # Evaluasi model
     mae = mean_absolute_error(y_test, y_pred)
     mse = mean_squared_error(y_test, y_pred)
     r2 = r2_score(y_test, y_pred)

     print("Laporan Evaluasi:")
     print(f"Mean Absolute Error (MAE): {mae:.2f}")
     print(f"Mean Squared Error (MSE): {mse:.2f}")
     print(f"R² Score: {r2:.2f}")
```

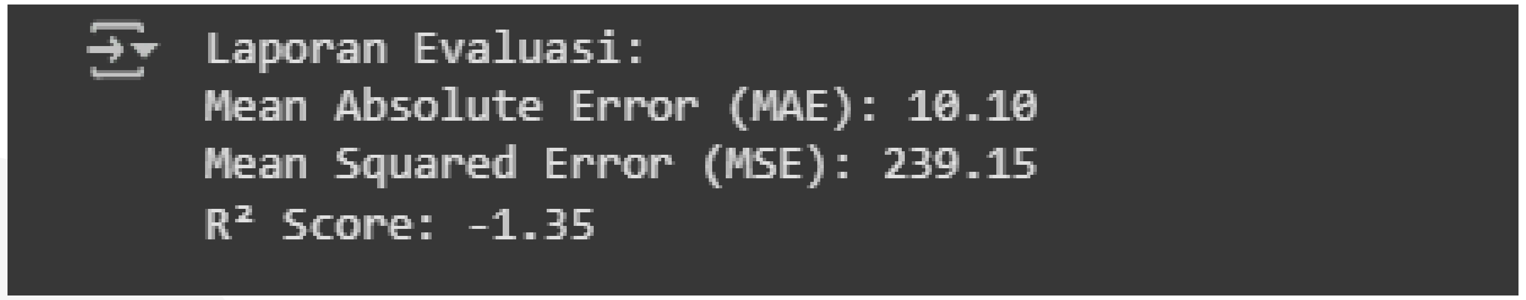
- Mean Absolute Error (MAE): The average of the absolute differences between actual and predicted values.
- Mean Squared Error (MSE): The average of the squared differences between actual and predicted values.
- R2 Score: A measure of how well the model explains the variance in the target variable (range from 0 to 1).

PREDICT AND EVALUATE

Model Evaluation



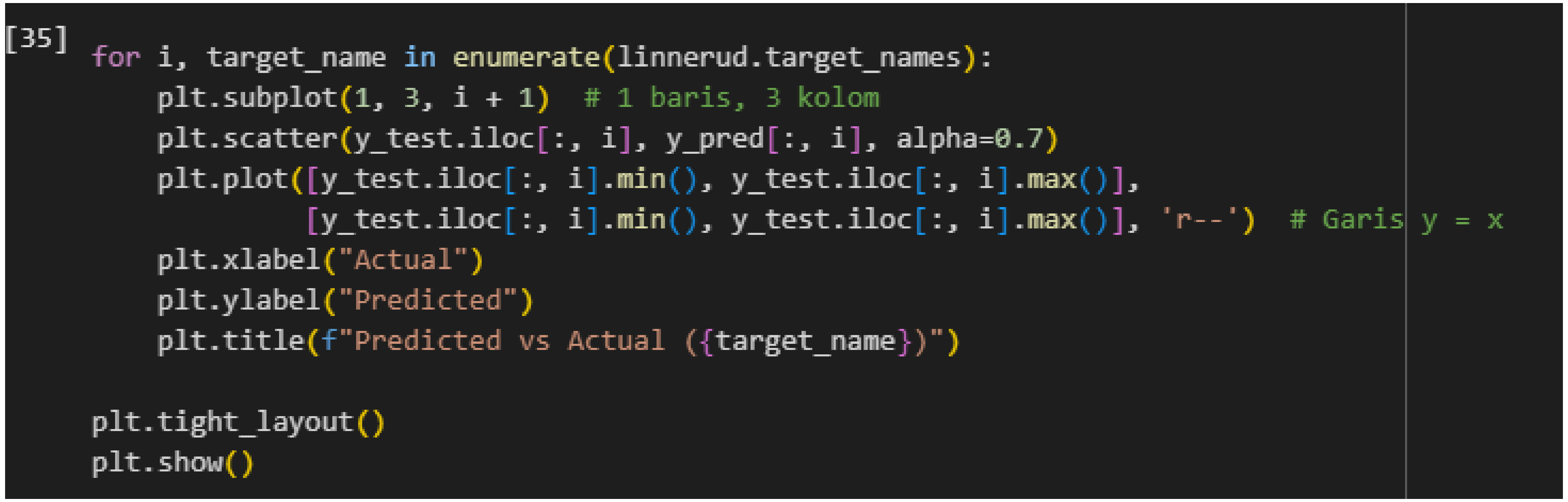
```
[31] y_pred = model.predict(X_test)

[34] # Evaluasi model
     mae = mean_absolute_error(y_test, y_pred)
     mse = mean_squared_error(y_test, y_pred)
     r2 = r2_score(y_test, y_pred)

     print("Laporan Evaluasi:")
     print(f"Mean Absolute Error (MAE): {mae:.2f}")
     print(f"Mean Squared Error (MSE): {mse:.2f}")
     print(f"R² Score: {r2:.2f}")
```



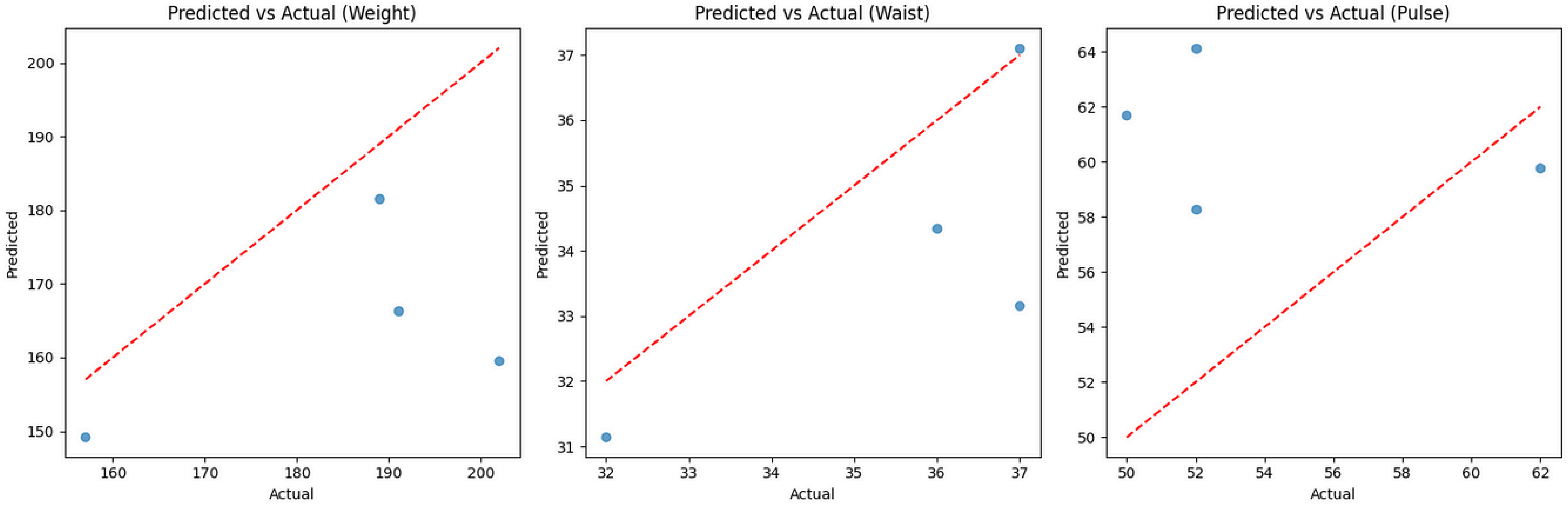
```
Laporan Evaluasi:
Mean Absolute Error (MAE): 10.10
Mean Squared Error (MSE): 239.15
R² Score: -1.35
```

- Mean Absolute Error (MAE): The average of the absolute differences between actual and predicted values.
- Mean Squared Error (MSE): The average of the squared differences between actual and predicted values.
- R2 Score: A measure of how well the model explains the variance in the target variable (range from 0 to 1).

PREDICT AND EVALUATE



```
[35]
for i, target_name in enumerate(linnerud.target_names):
    plt.subplot(1, 3, i + 1)  # 1 baris, 3 kolom
    plt.scatter(y_test.iloc[:, i], y_pred[:, i], alpha=0.7)
    plt.plot([y_test.iloc[:, i].min(), y_test.iloc[:, i].max()],
             [y_test.iloc[:, i].min(), y_test.iloc[:, i].max()], 'r--')  # Garis y = x
    plt.xlabel("Actual")
    plt.ylabel("Predicted")
    plt.title(f"Predicted vs Actual ({target_name})")

plt.tight_layout()
plt.show()
```



- For each target variable, a scatter plot is created that compares the actual values (y_test) to the predicted values (y_pred).
- A red dashed line (y=x) is drawn to indicate where perfect predictions would lie.
- The plots are displayed for each target variable in the dataset.

THANK YOU

🔗 www.linkedin.com/in/dewiyuliana1507

✉ dewiyulianaa938@gmail.com

📷 @dhyli_ana