JOBSHEET 13

Multiple Linear Regression

Pendahuluan

Modul ini menjelaskan penerapan algoritma Multiple Linear Regression dengan menggunakan studi kasus Prediksi Harga Rumah yang dilengkapi dengan tahapan yang bisa diambil kesimpulannya

Tujuan Praktikum

Setelah menyelesaikan praktikum ini, mahasiswa mampu:

- Membangun model regresi linier berganda menggunakan OLS
- Memvisualisasikan dan menginterpretasikan hasil model
- Melakukan validasi model dan uji asumsi klasik

Peralatan yang dibutuhkan

Beberapa peralatan yang dibutuhkan dalam menyelesaikan praktikum ini adalah:

- Aplikasi Microsoft Excel
- Google Colab
- Google Drive
- Koneksi Internet
- Browser Web

Praktikum

Praktikum Simple Linear Regression

Studi Kasus yang digunakan dalam praktikum ini adalah data pada tabel sebagai berikut:

No	Luas Tanah (m^2)	Kamar Tidur	Jarak Ke Pusat (km)	Usia Bangunan (tahun)	Harga (juta)
1	100	2	10	5	500
2	150	3	9	10	600
3	200	3	8	15	650
4	250	4	7	20	700
5	300	4	6	5	750
6	350	5	5	10	800
7	400	5	4	15	850
8	450	6	3	20	900
G	500	6	2	25	950
10	550	7	1	30	1000

Lakukan praktikum sesuai tahapan berikut:

a. Buka aplikasi web browser

b. Buka Google Colabs dan berikan nama file “MultipleLinearRegression.ipynb”

c. Lakukan Import Library yang dibutuhkan

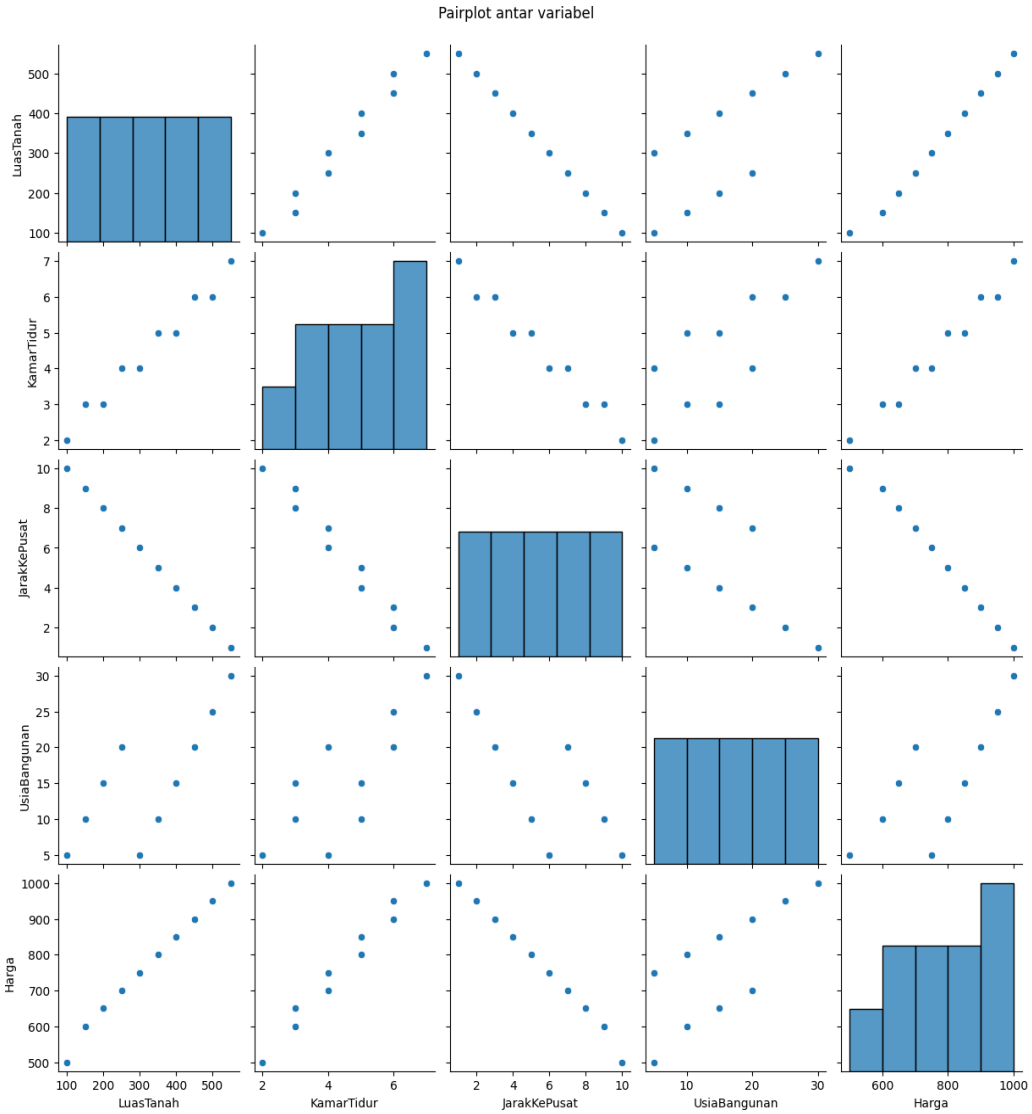
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import scipy.stats as stats
```

d. Membangun dataset simulasi

```
data = pd.DataFrame({
    'LuasTanah': [100, 150, 200, 250, 300, 350, 400, 450, 500, 550],
    'KamarTidur': [2, 3, 3, 4, 4, 5, 5, 6, 6, 7],
    'JarakKePusat': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1],
    'UsiaBangunan': [5, 10, 15, 20, 5, 10, 15, 20, 25, 30],
    'Harga': [500, 600, 650, 700, 750, 800, 850, 900, 950, 1000]
})
```

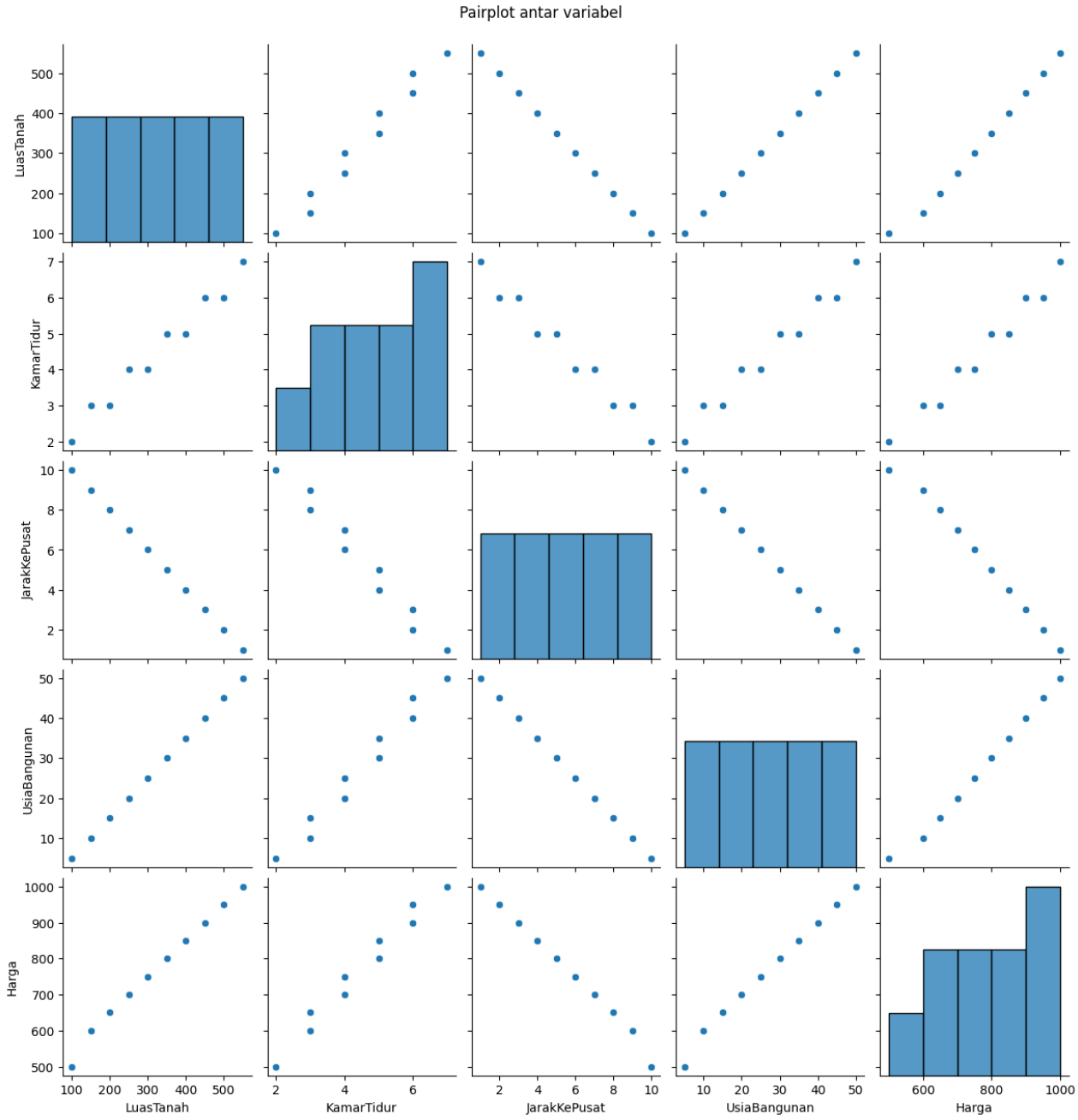
e. Lakukan visualisasi terhadap dataset simulasi dengan hubungan antar variable

```
sns.pairplot(data)
plt.suptitle("Pairplot antar variabel", y=1.02)
plt.show()
```



Percobaan yang saya lakukan

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import scipy.stats as stats

data = pd.DataFrame({
    'LuasTanah': [100, 150, 200, 250, 300, 350, 400, 450, 500, 550],
    'KamarTidur': [2, 3, 3, 4, 4, 5, 5, 6, 6, 7],
    'JarakKePusat': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1],
    'UsiaBangunan': [5, 10, 15, 20, 25, 30, 35, 40, 45, 50],
    'Harga': [500, 600, 650, 700, 750, 800, 850, 900, 950, 1000]
})

sns.pairplot(data)
plt.suptitle("Pairplot antar variabel", y=1.02)
plt.show()
```



f. Lakukan persiapan data

```
X = data[['LuasTanah', 'KamarTidur', 'JarakKePusat', 'UsiaBangunan']]
y = data['Harga']
```

g. Lakukan split data training dan data testing

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
X_train_const = sm.add_constant(X_train)
X_test_const = sm.add_constant(X_test)
```

h. Bangun Model OLS

```
model = sm.OLS(y_train, X_train_const).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Harga   R-squared:                       0.993
Model:                            OLS   Adj. R-squared:                  0.985
Method:                 Least Squares   F-statistic:                     133.9
Date:                Mon, 19 May 2025   Prob (F-statistic):            0.00108
Time:                        03:59:34   Log-Likelihood:                -28.080
No. Observations:                   7   AIC:                             64.16
Df Residuals:                       3   BIC:                             63.94
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.9557      0.334      8.847      0.003       1.893       4.019
LuasTanah      1.8961      0.598      3.171      0.050      -0.007       3.799
KamarTidur   -12.5000     52.540     -0.238      0.827    -179.705     154.705
JarakKePusat  35.4308      3.998      8.862      0.003      22.707      48.155
UsiaBangunan   0.6250      1.943      0.322      0.769      -5.559       6.809
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   1.750
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.767
Skew:                        -0.702   Prob(JB):                        0.682
Kurtosis:                     2.187   Cond. No.                     8.00e+18
==============================================================================
```

- R-squared: seberapa baik model menjelaskan variasi Harga
- P>|t|: nilai signifikansi (variabel signifikan jika < 0.05)
- Koefisien: perubahan rata-rata Harga tiap 1 unit perubahan variabel

Percobaan yang saya lakukan

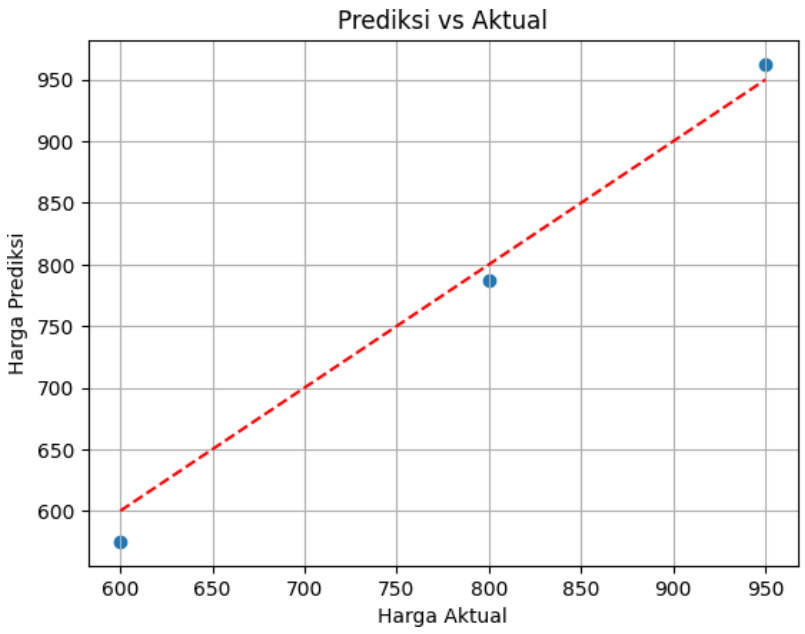
```
X = data[['LuasTanah', 'KamarTidur', 'JarakKePusat', 'UsiaBangunan']]
y = data['Harga']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
X_train_const = sm.add_constant(X_train)
X_test_const = sm.add_constant(X_test)

model = sm.OLS(y_train, X_train_const).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Harga   R-squared:                       0.992
Model:                            OLS   Adj. R-squared:                  0.988
Method:                 Least Squares   F-statistic:                     258.7
Date:                Thu, 22 May 2025   Prob (F-statistic):           5.88e-05
Time:                        01:20:22   Log-Likelihood:                -28.199
No. Observations:                   7   AIC:                             62.40
Df Residuals:                       4   BIC:                             62.24
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.4771      0.204     12.133      0.000       1.910       3.044
LuasTanah      2.8892      0.497      5.810      0.004       1.508       4.270
KamarTidur    -0.8621     33.554     -0.026      0.981     -94.024      92.300
JarakKePusat  29.6672      2.441     12.154      0.000      22.890      36.444
UsiaBangunan -12.0965      0.976    -12.392      0.000     -14.807      -9.386
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   1.655
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.594
Skew:                        -0.639   Prob(JB):                        0.743
Kurtosis:                     2.362   Cond. No.                     5.30e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.12e-32. This might indicate that there are
```

i. Lakukan prediksi dan Evaluasi

```
y_pred = model.predict(X_test_const)
```

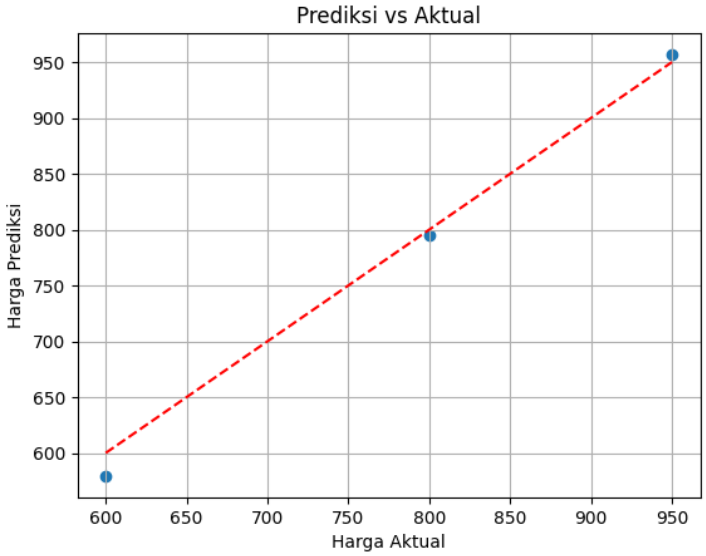
```
plt.scatter(y_test, y_pred)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel("Harga Aktual")
plt.ylabel("Harga Prediksi")
plt.title("Prediksi vs Aktual")
plt.grid(True)
plt.show()
```



- Titik-titik mendekati garis merah → prediksi akurat
- Jarak jauh → model belum optimal

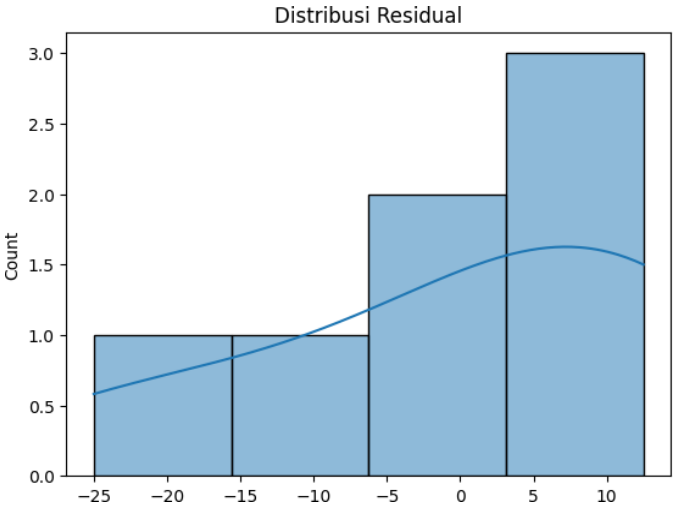
Percobaan yang saya lakukan

```
y_pred = model.predict(X_test_const)

plt.scatter(y_test, y_pred)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel("Harga Aktual")
plt.ylabel("Harga Prediksi")
plt.title("Prediksi vs Aktual")
plt.grid(True)
plt.show()
```



j. Lakukan Validasi Asumsi Model (Normalitas Residual)

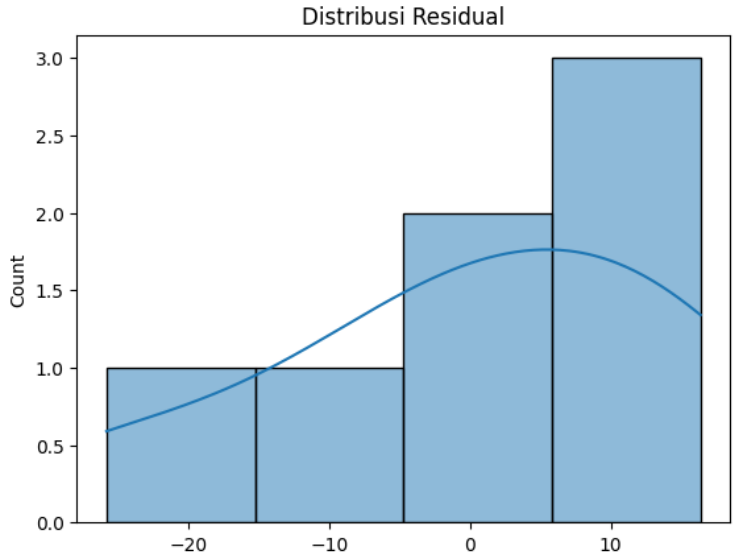
```
residuals = model.resid
sns.histplot(residuals, kde=True)
plt.title("Distribusi Residual")
plt.show()

print("Shapiro-Wilk:", stats.shapiro(residuals))
```



- Histogram menyerupai kurva normal → asumsi terpenuhi
- Shapiro-Wilk p > 0.05 → residual normal

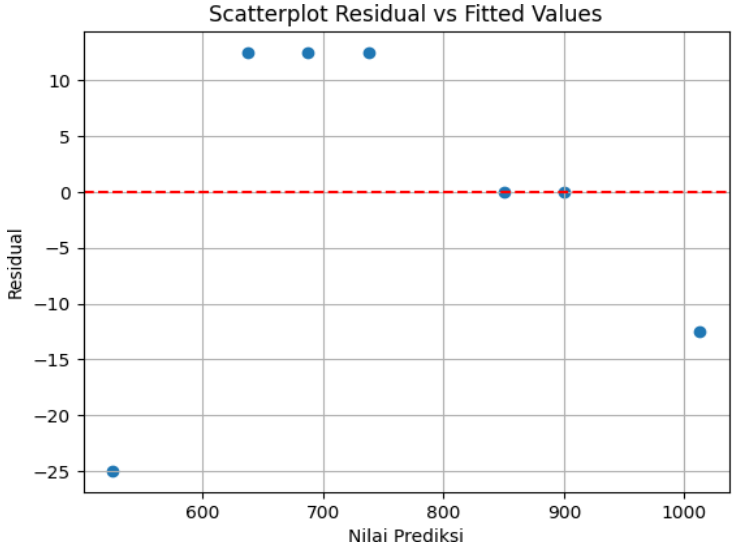
Percobaan yang saya lakukan

```
residuals = model.resid
sns.histplot(residuals, kde=True)
plt.title("Distribusi Residual")
plt.show()

print("Shapiro-Wilk:", stats.shapiro(residuals))
```



k. Lakukan Validasi Asumsi Model (Homoskedastisitas)

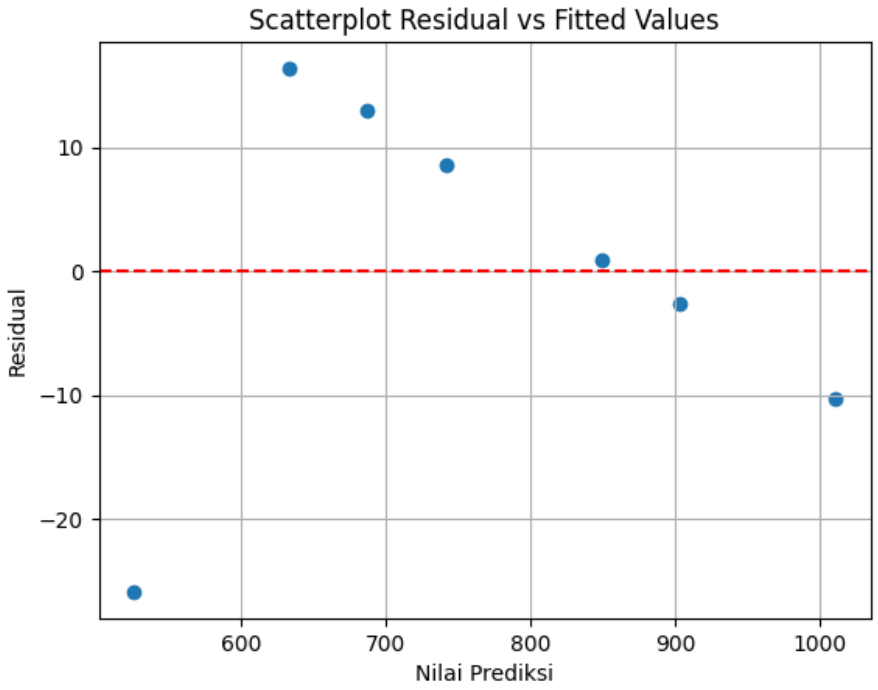
```
plt.scatter(model.fittedvalues, residuals)
plt.axhline(0, color='red', linestyle='--')
plt.title("Scatterplot Residual vs Fitted Values")
plt.xlabel("Nilai Prediksi")
plt.ylabel("Residual")
plt.grid(True)
plt.show()
```



- Jika pola menyebar acak dan simetris → tidak terjadi heteroskedastisitas

Percobaan yang saya lakukan

```
plt.scatter(model.fittedvalues, residuals)
plt.axhline(0, color='red', linestyle='--')
plt.title("Scatterplot Residual vs Fitted Values")
plt.xlabel("Nilai Prediksi")
plt.ylabel("Residual")
plt.grid(True)
plt.show()
```



I. Lakukan Validasi Asumsi Model (Multikolinearitas (VIF))

```
vif = pd.DataFrame()
vif["feature"] = X_train.columns
vif["VIF"] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape[1])]
print(vif)
```

	feature	VIF
0	LuasTanah	745.903158
1	KamarTidur	117.357143
2	JarakKePusat	10.697428
3	UsiaBangunan	4.272321

- VIF < 10 → tidak ada multikolinearitas
- VIF > 10 → ada korelasi tinggi antar variabel independen → pertimbangkan menghapusnya

Percobaan yang saya lakukan

```
vif = pd.DataFrame()
vif["feature"] = X_train.columns
vif["VIF"] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape[1])]
print(vif)
```

	feature	VIF
0	LuasTanah	inf
1	KamarTidur	61.694581
2	JarakKePusat	inf
3	UsiaBangunan	inf

LATIHAN

1. Tambahkan variabel AksesTransportasi (skor 1–10) sebagai variabel baru.
2. Bangun ulang model dan ulangi semua proses:
   a. Analisis hubungan variabel baru
   b. Uji signifikansi
   c. Visualisasi dan interpretasi ulang

Jawaban

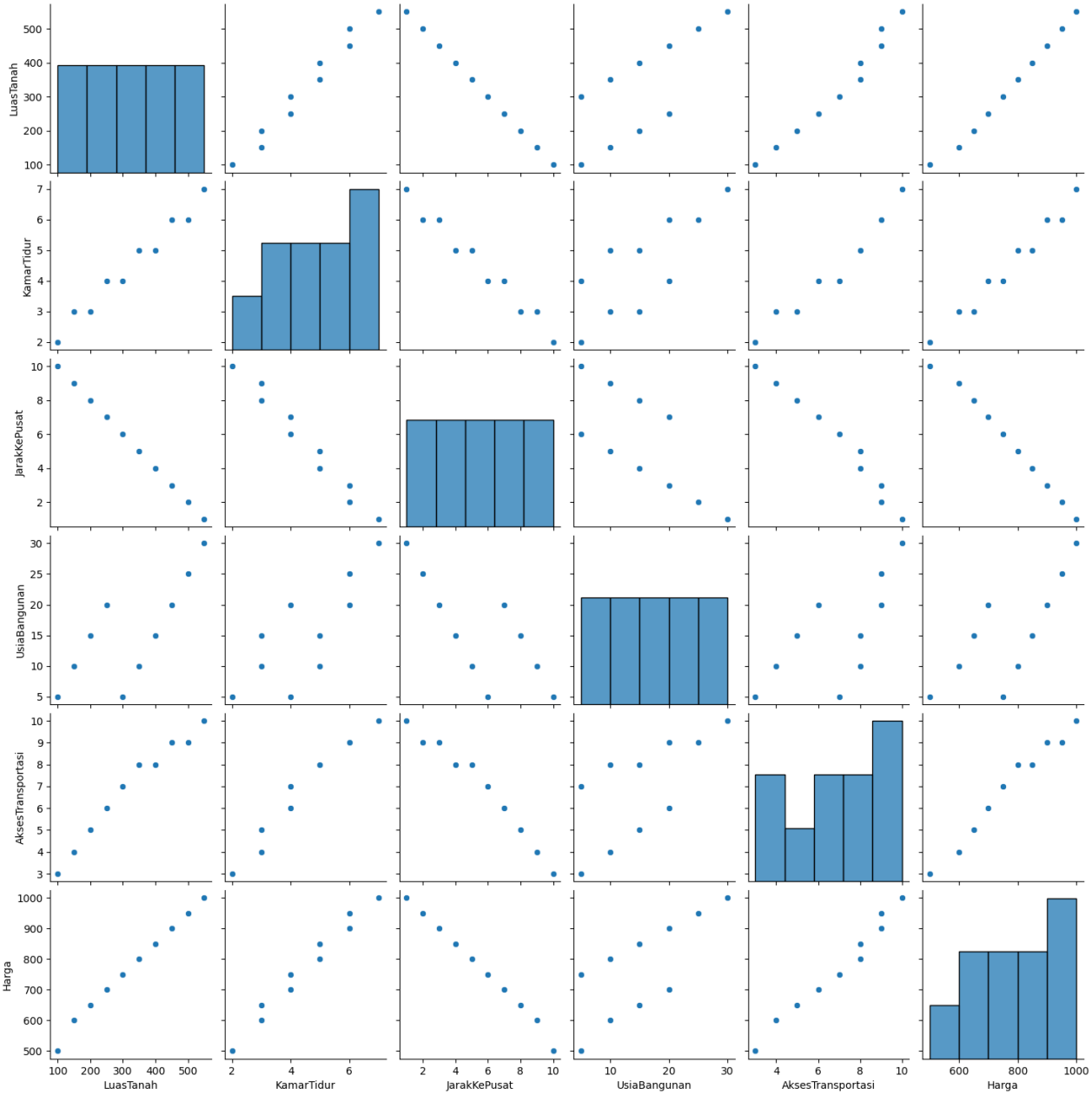
1. Menambahkan variabel AksesTransportasi

```
data = {
    'LuasTanah': [100, 150, 200, 250, 300, 350, 400, 450, 500, 550],
    'KamarTidur': [2, 3, 3, 4, 4, 5, 5, 6, 6, 7],
    'JarakKePusat': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1],
    'UsiaBangunan': [5, 10, 15, 20, 5, 10, 15, 20, 25, 30],
    'AksesTransportasi': [3, 4, 5, 6, 7, 8, 8, 9, 9, 10],
    'Harga': [500, 600, 650, 700, 750, 800, 850, 900, 950, 1000]
}
```

2. Mengulangi semua proses

Import Library dan bangun dataset simulasi

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import scipy.stats as stats

data = pd.DataFrame({
    'LuasTanah': [100, 150, 200, 250, 300, 350, 400, 450, 500, 550],
    'KamarTidur': [2, 3, 3, 4, 4, 5, 5, 6, 6, 7],
    'JarakKePusat': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1],
    'UsiaBangunan': [5, 10, 15, 20, 25, 30, 35, 40, 45, 50],
    'AksesTransportasi': [3, 4, 5, 6, 7, 8, 8, 9, 9, 10],
    'Harga': [500, 600, 650, 700, 750, 800, 850, 900, 950, 1000]
})

sns.pairplot(data)
plt.suptitle("Pairplot antar variabel", y=1.02)
plt.show()
```



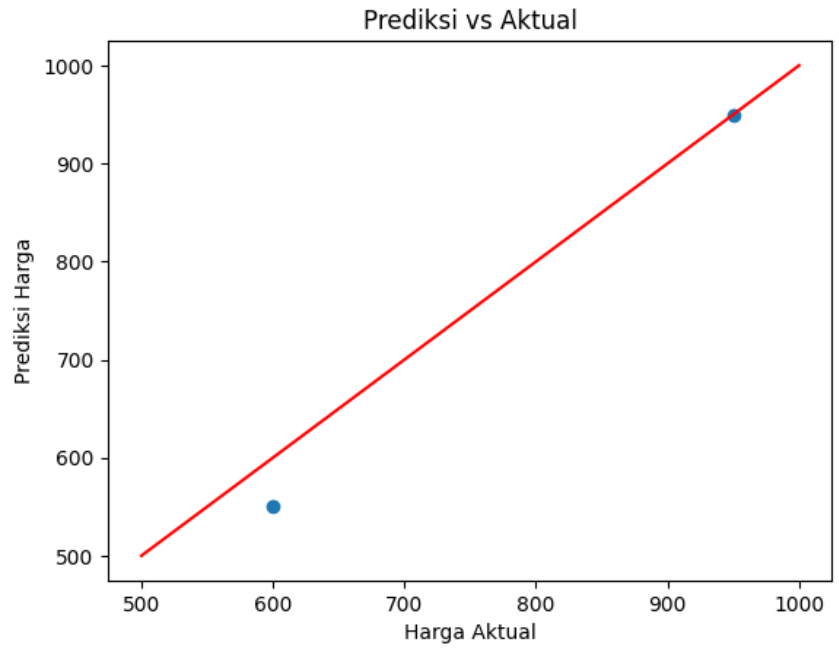
Visualisasi korelasi antar variabel serta menambahkan intercept dan bangun model OLS

```
X = data[['LuasTanah', 'KamarTidur', 'JarakKePusat', 'UsiaBangunan']]
y = data['Harga']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
X_train_const = sm.add_constant(X_train)
X_test_const = sm.add_constant(X_test)

model = sm.OLS(y_train, X_train_const).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Harga   R-squared:                       1.000
Model:                            OLS   Adj. R-squared:                  1.000
Method:                 Least Squares   F-statistic:                 4.851e+28
Date:                Thu, 22 May 2025   Prob (F-statistic):           1.52e-43
Time:                        01:59:45   Log-Likelihood:                 214.16
No. Observations:                   8   AIC:                            -418.3
Df Residuals:                       3   BIC:                            -417.9
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
const              2.5022   1.57e-14   1.59e+14      0.000       2.502       2.502
LuasTanah          1.3500   2.33e-14   5.78e+13      0.000       1.350       1.350
KamarTidur       -50.0000   2.19e-12  -2.28e+13      0.000     -50.000     -50.000
JarakKePusat      29.9998   1.88e-13   1.59e+14      0.000      30.000      30.000
UsiaBangunan       2.5000   8.92e-14    2.8e+13      0.000       2.500       2.500
AksesTransportasi 50.0000   1.29e-12   3.87e+13      0.000      50.000      50.000
==============================================================================
Omnibus:                        1.118   Durbin-Watson:                   0.552
Prob(Omnibus):                  0.572   Jarque-Bera (JB):                0.736
Skew:                          -0.429   Prob(JB):                        0.692
Kurtosis:                       1.786   Cond. No.                     4.21e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 5.6e-32. This might indicate that there are
```

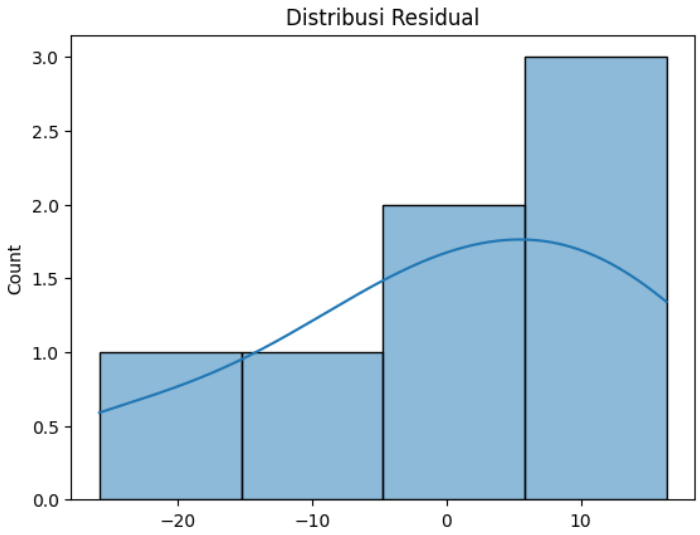
Prediksi dan evaluasi

```
plt.scatter(y_test, y_pred)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel("Harga Aktual")
plt.ylabel("Harga Prediksi")
plt.title("Prediksi vs Aktual")
plt.grid(True)
plt.show()
```



Validasi Normalitas Residual

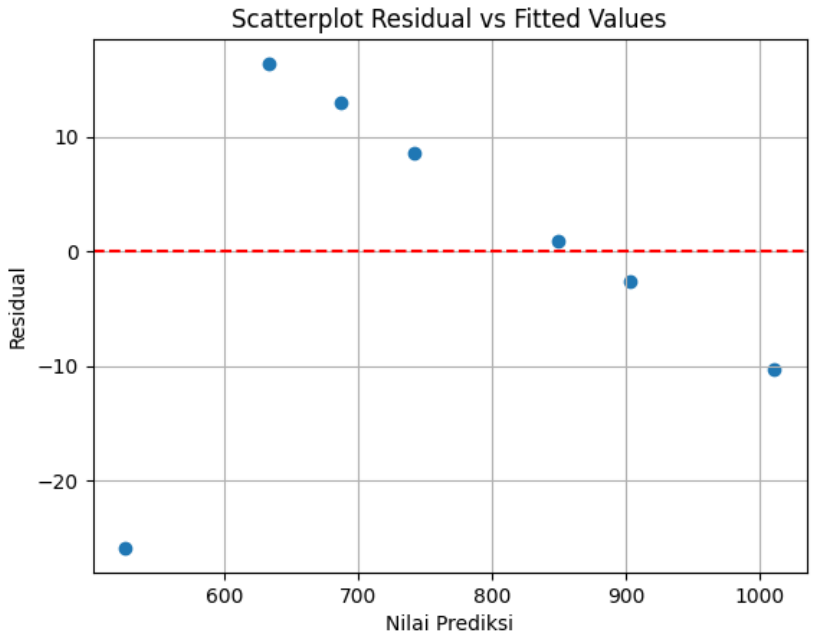
```
residuals = model.resid
sns.histplot(residuals, kde=True)
plt.title("Distribusi Residual")
plt.show()
```



Shapiro-Wilk: ShapiroResult(statistic=np.float64(0.9449282161167907), pvalue=np.float64(0.6834282320464362))

Validasi Homoskedastisitas

```
plt.scatter(model.fittedvalues, residuals)
plt.axhline(0, color='red', linestyle='--')
plt.title("Scatterplot Residual vs Fitted Values")
plt.xlabel("Nilai Prediksi")
plt.ylabel("Residual")
plt.grid(True)
plt.show()
```



Validasi Multikolinearitas (VIF)

```
vif = pd.DataFrame()
vif["feature"] = X_train.columns
vif["VIF"] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape[1])]
print(vif)
```

	feature	VIF
0	LuasTanah	inf
1	KamarTidur	61.694581
2	JarakKePusat	inf
3	UsiaBangunan	inf