



DigitalSkola
Uncover The World of Digital Skills with Us

FINAL PROJECT - DIAMOND PRICE PREDICTION

Data Science Batch 36



Insight Squad



OUR TEAM'S



HABIBAH DIAN K.



IGOH SATRIA F. P.



IRDAM ELBA S.



MARCHO TUMBADE



WISNU SETYA W.



MUSLIH



TABLE OF CONTENT



- 1 About Data**
- 2 EDA (Exploratory Data Analysis)**
- 3 PreProcessing**
- 4 Feature Engineering**
- 5 Modelling**

ABOUT DATA

ABOUT DATA

Tentang Data

Dataset diamond berisi beberapa atribut dan harga dengan total data 53940 berlian. Terdapat 11 atribut yang disertakan dalam dataset termasuk target yaitu harga.

- **carat (0.2-5.01):** Carat adalah berat fisik berlian yang diukur dalam karat.
- **cut (Ideal, Premium, Good, Very Good, Fair):** Kualitas potongan. Semakin presisi potongan berlian, semakin baik berlian tersebut, sehingga memiliki grade yang tinggi.
- **color:** warna berlian : dari D = Terbaik sampai J = Terburuk: Warna berlian kualitas permata hadir dalam banyak nuansa. Dalam rentang dari tidak berwarna hingga kuning muda atau coklat muda. Berlian tidak berwarna adalah yang paling langka. Warna alami lainnya (biru, merah, merah muda misalnya) dikenal sebagai "fancy," dan penilaian warnanya berbeda dari berlian putih tidak berwarna.
- **clarity (I1 (terburuk), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (terbaik)):** Berlian dapat memiliki karakteristik internal yang dikenal sebagai inklusi atau karakteristik eksternal yang dikenal sebagai noda. Berlian tanpa inklusi atau noda sangat langka; namun, sebagian besar karakteristik hanya dapat dilihat dengan pembesaran.



ABOUT DATA

Tentang Data

- **depth (43-79)** merupakan **tinggi berlian**: Ini adalah **persentase total kedalaman yang sama dengan z / mean(x, y) = 2 * z / (x + y)**. Kedalaman berlian adalah tinggi berlian (dalam milimeter) yang diukur dari culet (ujung bawah) hingga table (permukaan atas datar).
- **table (43-95)**: Ini adalah **lebar bagian ukuran atas berlian relatif terhadap titik terlebar**.
- **price dalam dolar AS (326-18,823)**: Ini adalah **harga berlian dalam dolar AS**.
- **x (0 - 10.74)**: **Panjang berlian (mm)**
- **y (0 - 58.9)**: **Lebar berlian (mm) berdasarkan dataset**
- **z (0 - 31.8)**: **Kedalaman berlian (mm) berdasarkan dataset**

```
[ ] data.head(10)
```

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	F	VS1	56.9	65.0	327	4.05	4.07	2.31

EDA (EXPLORATORY DATA ANALYSIS)

EDA (EXPLORATORY DATA ANALYSIS)

1. Descriptive Statistics:

Output: Tabel menampilkan statistik deskriptif seperti count, mean, std, min, 25%, 50%, 75%, dan max untuk setiap fitur dalam dataset.

Penjelasan: Statistik deskriptif ini memberikan gambaran umum mengenai distribusi data.

2. Missing Values:

Output: Tidak ada missing values di dataset. Hal ini dapat dilihat dari hasil data.info() dan data.isna().sum() yang menunjukkan semua kolom memiliki jumlah entri yang sama dan nilai False untuk pengecekan missing values.

Penjelasan: Pengecekan ini penting untuk memastikan tidak ada data yang hilang yang dapat mempengaruhi hasil analisis dan model.

3. Data Types and Shape:

Output: Informasi tipe data dari setiap kolom (data.dtypes) dan ukuran dataset (data.shape). Dataset memiliki 53940 entri dan 10 kolom setelah menghapus kolom yang tidak diperlukan.

Penjelasan: Mengetahui tipe data dan ukuran dataset membantu dalam proses pembersihan dan persiapan data untuk analisis lebih lanjut. Tipe data juga penting dalam menentukan metode analisis yang sesuai.

4. Data Cleaning:

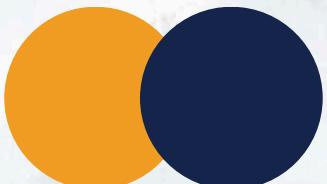
Output: Kolom 'Unnamed: 0' dihapus karena tidak diperlukan, dan beberapa kolom diubah namanya untuk kemudahan analisis (misalnya, x menjadi panjang_berlian).

Penjelasan: Langkah ini dilakukan untuk membersihkan dataset dari kolom yang tidak relevan dan membuat nama kolom lebih deskriptif, sehingga memudahkan interpretasi dan analisis data.

EDA (EXPLORATORY DATA ANALYSIS)

```
[ ] # Pengecekan apakah terdapat data duplicate dalam dataset atau tidak  
duplicate_rows = data.duplicated()  
num_duplicate_rows = duplicate_rows.sum()  
print(f"Jumlah data duplikat : {num_duplicate_rows}")  
  
→ Jumlah data duplikat : 146
```

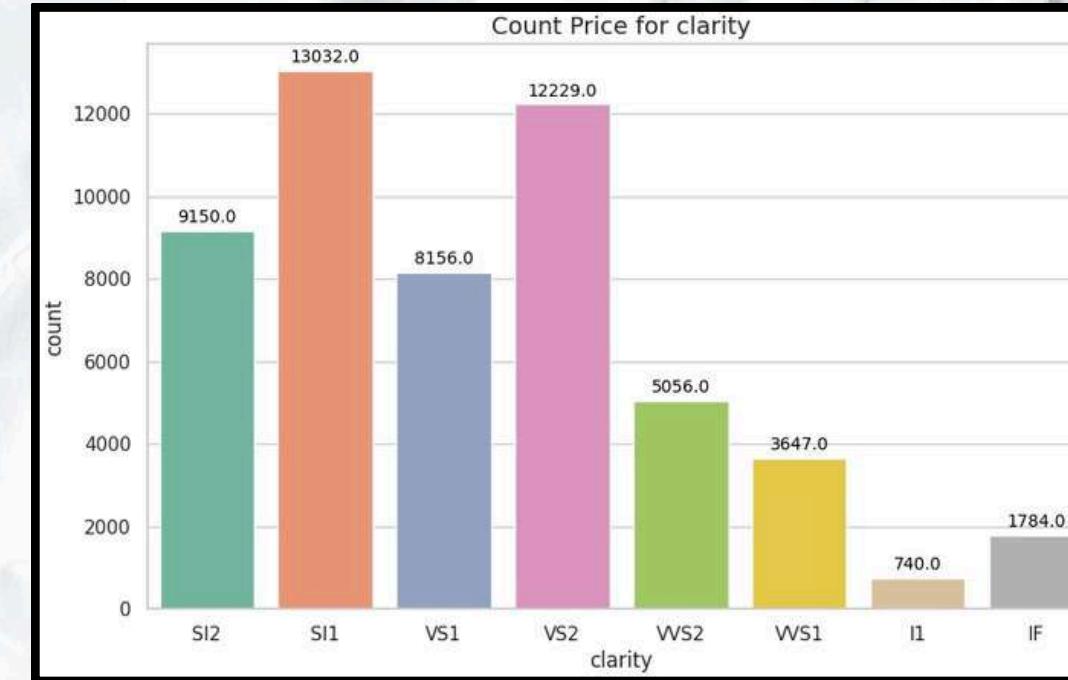
Setelah melakukan pengecekan data terkait data duplikasi ternyata terdapat 146 data duplikat dalam dataset yang dimiliki, sehingga langkah yang diambil ialah melakukan penghapusan data duplikat dengan alasan bahwa memastikan nantinya model tidak bias dan akurat dengan pertimbangan berdasarkan data yang akan digunakan ketika melakukan prediksi.



EDA (EXPLORATORY DATA ANALYSIS)



Dari antara cut(potongan berlian), cut 'ideal' merupakan jumlah price tertinggi, kemudian 'fair' terendah



Berlian dengan clarity atau kejernihan label 'I1' (terburuk) total price sangat jarang

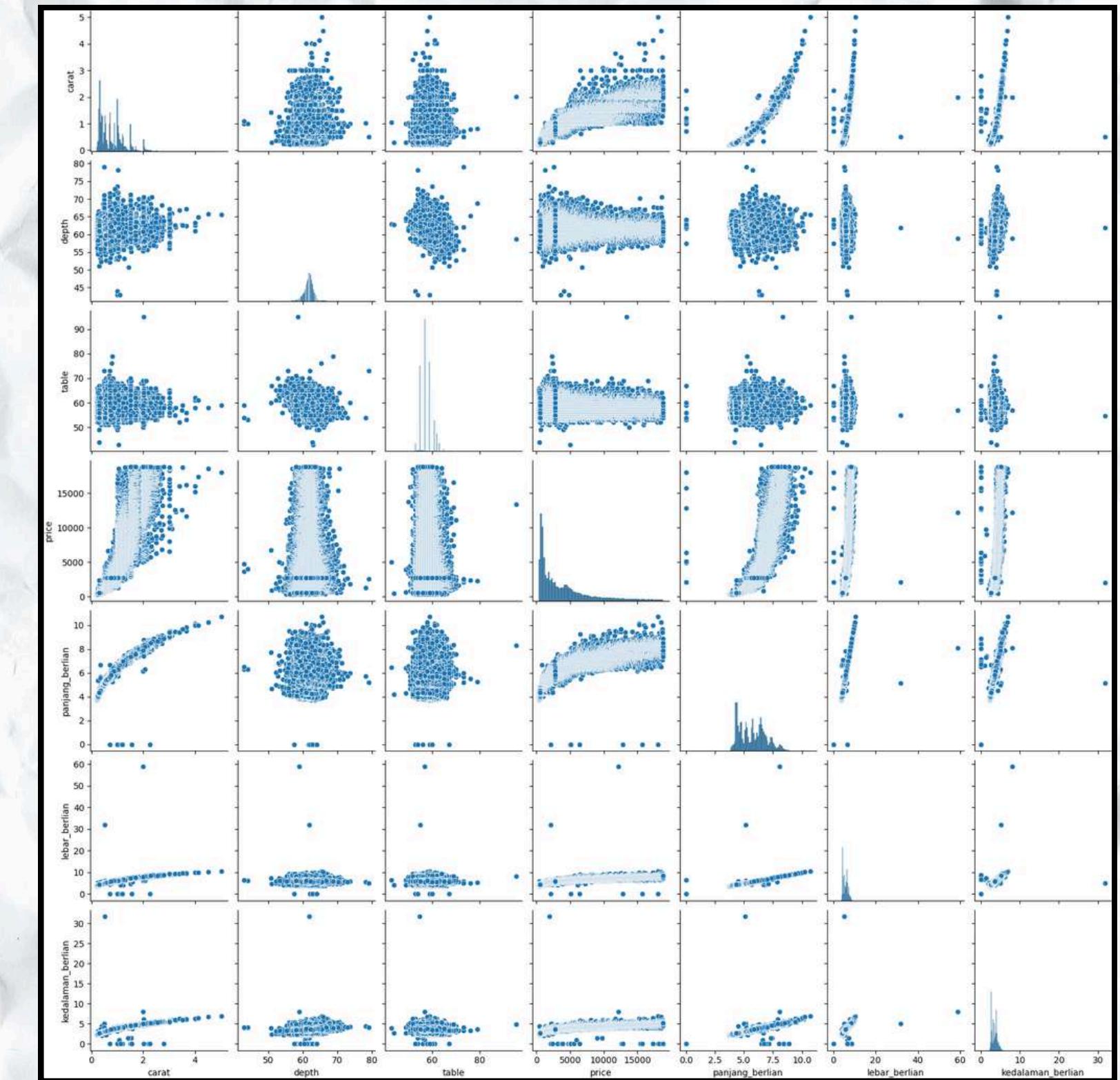


Berlian dengan warna J (terburuk) paling sedikit peminat atau dengan kategori price paling rendah

EDA (EXPLORATORY DATA ANALYSIS)

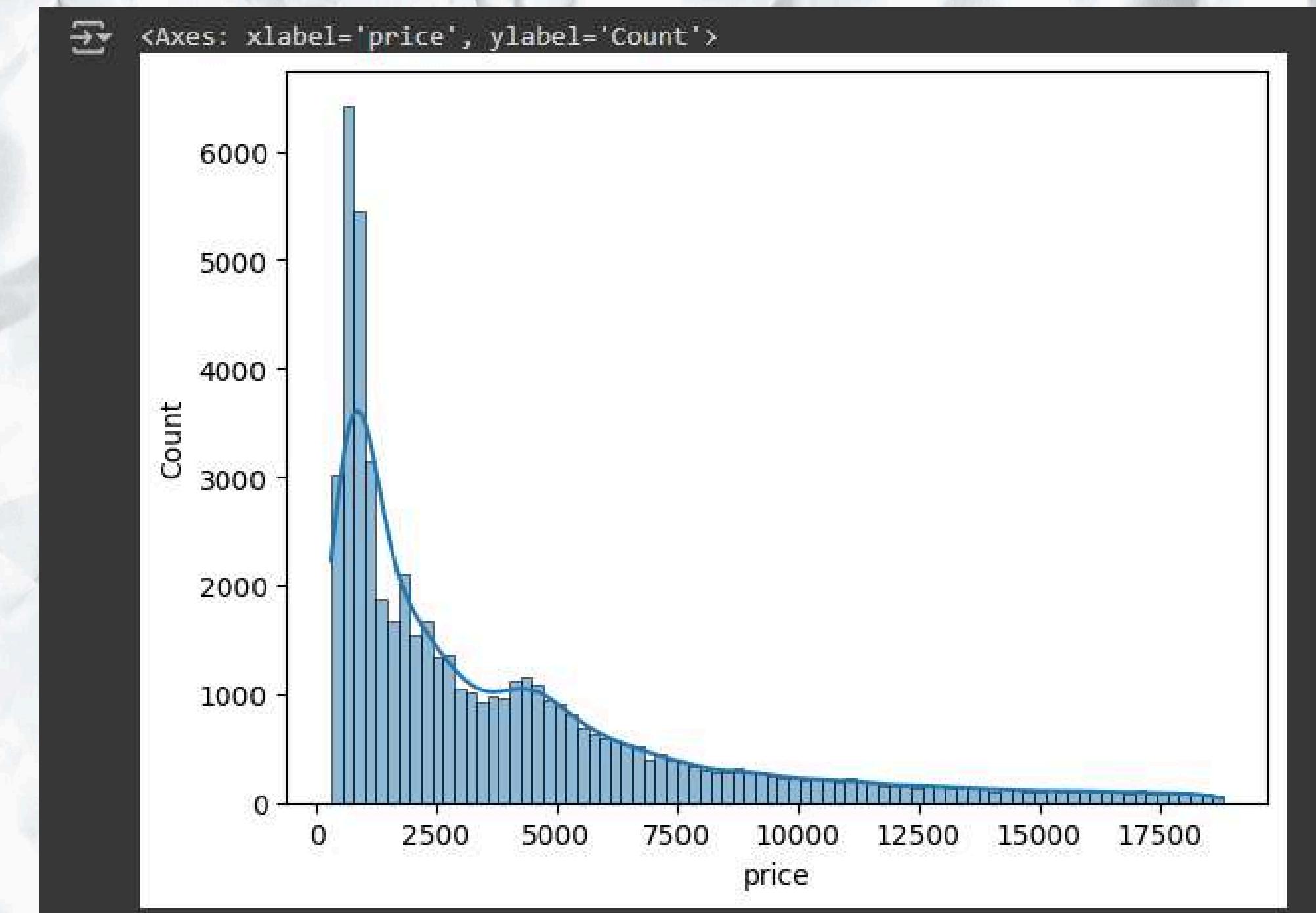
```
▶ # Merupakan sebaran variabel dalam bentuk visualisasi hubungan antar variabel  
sns.pairplot(data_clean, palette=cols)  
plt.show()
```

- 1. Dari visualisasi diatas dapat disimpulkan terdapat beberapa outliers (titik yang jauh) yang perlu ditangani karena bisa mempengaruhi data**
- 2. Dikarenakan terdapat outliers perlu dilakukan pengolahan data bersama team**
- 3. Melakukan pengecekan variabel target yaitu price**



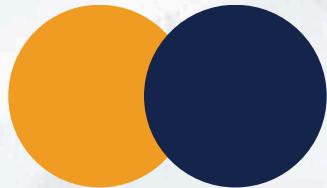
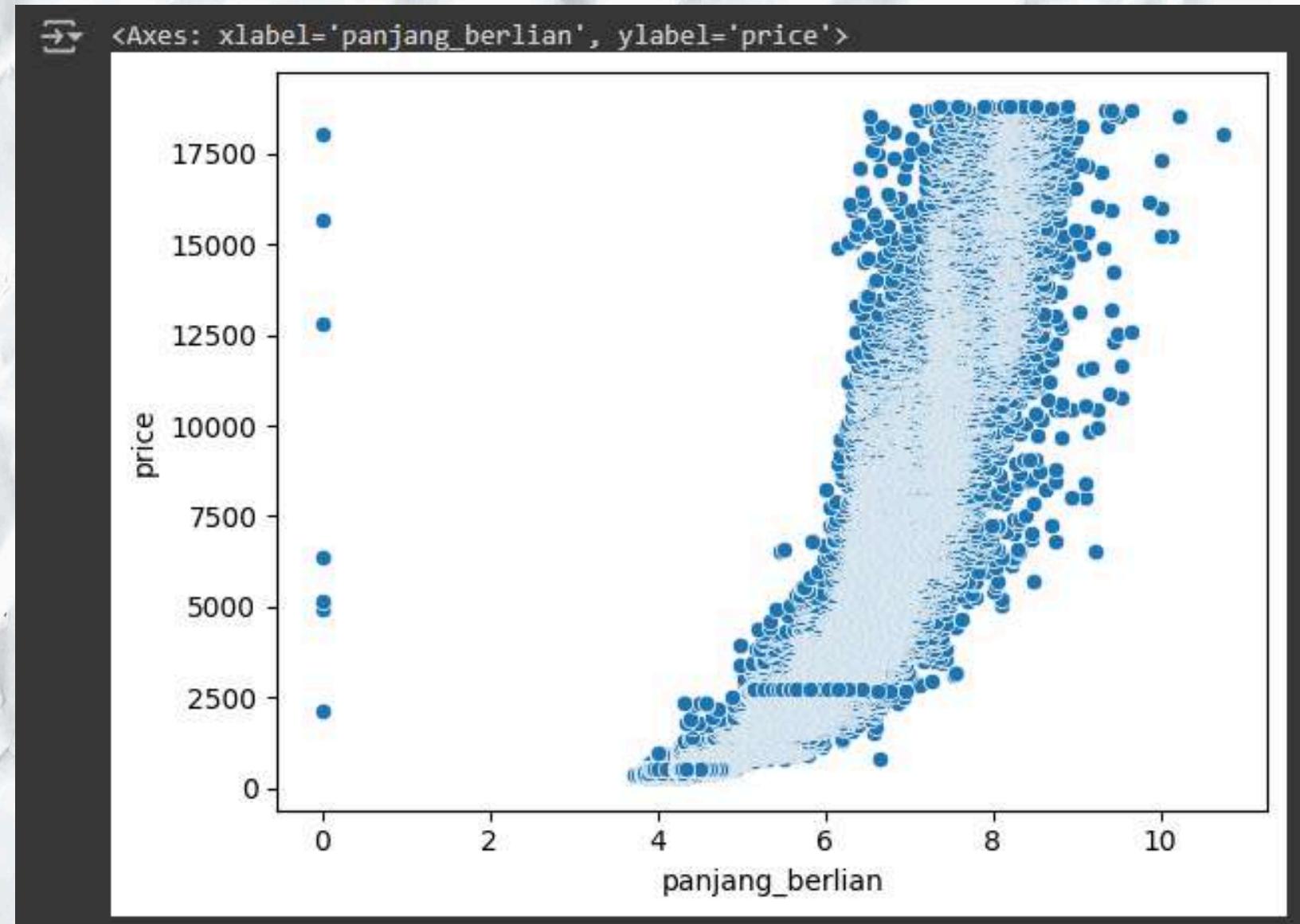
EDA (EXPLORATORY DATA ANALYSIS)

```
[ ] # Melihat Distribusi  
# Merupakan coloms target  
sns.histplot(data_clean['price'], kde=True)
```



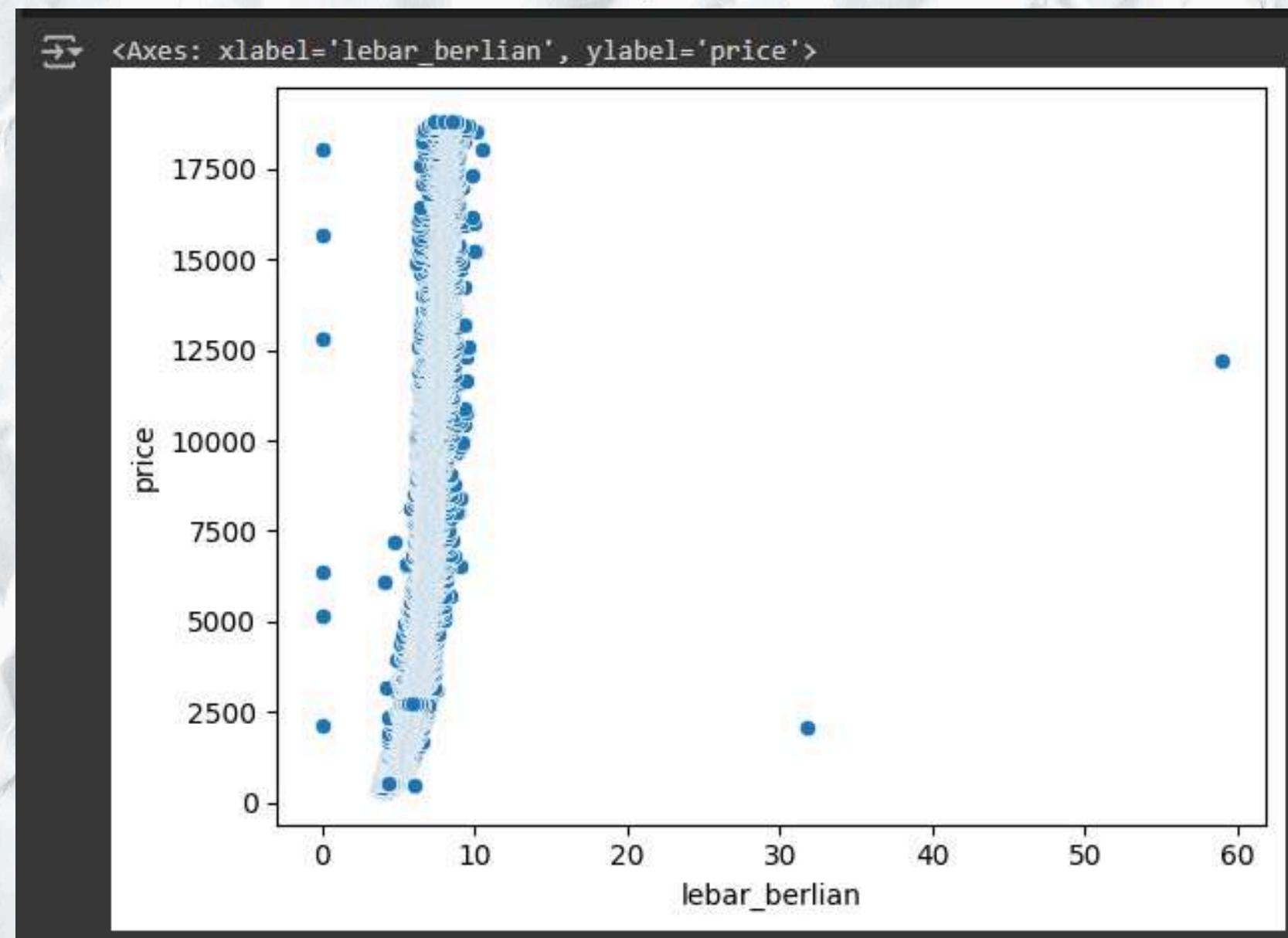
EDA (EXPLORATORY DATA ANALYSIS)

```
[ ] # Melakukan pengecekan outliers data berdasarkan colom "panjang berlian" berdasarkan price  
# Kesimpulan dari hasil visualization ialah tidak ada berlian yang tidak memiliki nilai panjang  
sns.scatterplot(x=data_clean['panjang_berlian'], y=data_clean['price'])
```



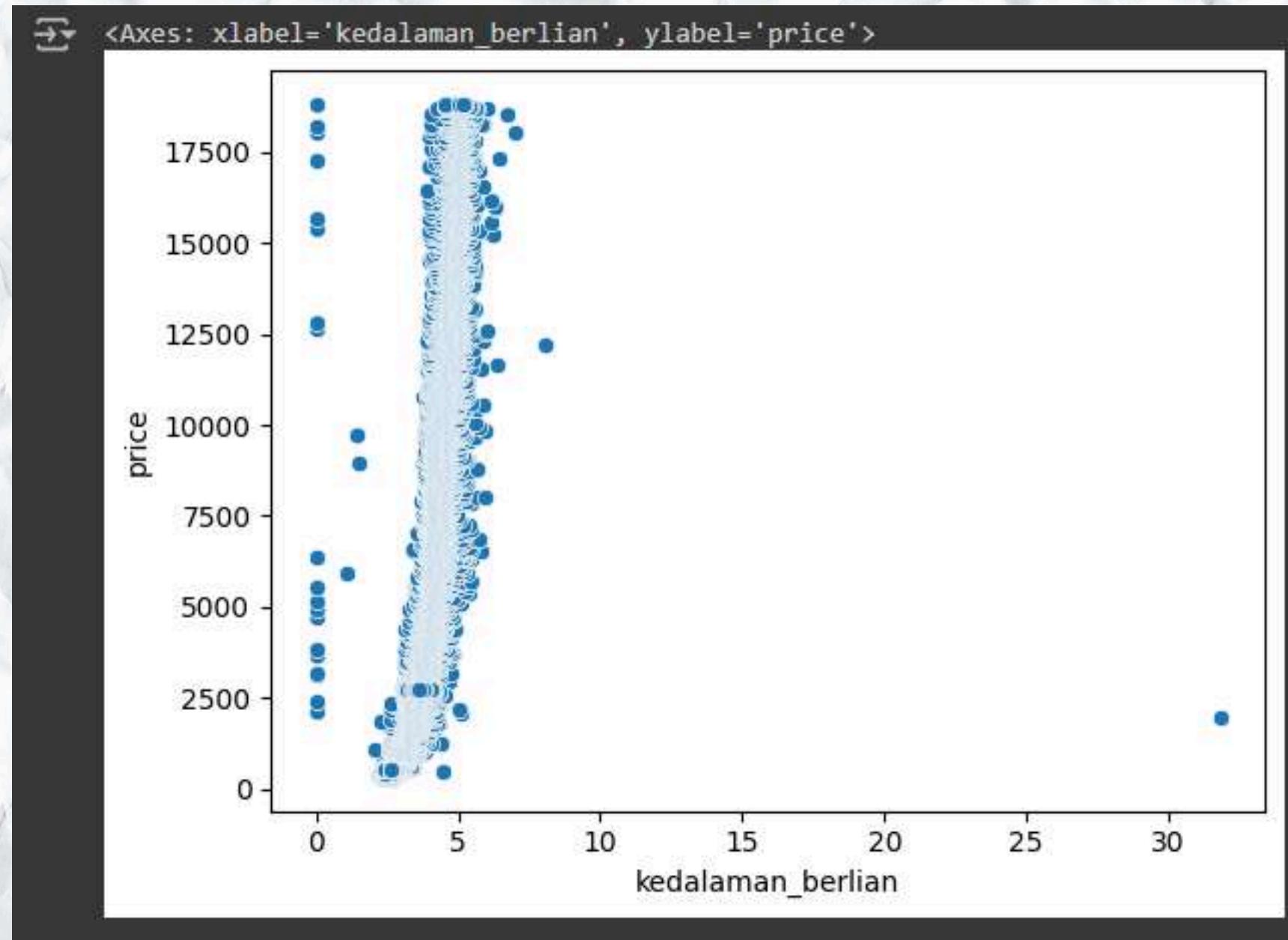
EDA (EXPLORATORY DATA ANALYSIS)

```
[ ] # Tidak ada berlian yang tidak memiliki ukuran lebar berlian  
sns.scatterplot(x=data_clean['lebar_berlian'], y=data_clean['price'])
```



EDA (EXPLORATORY DATA ANALYSIS)

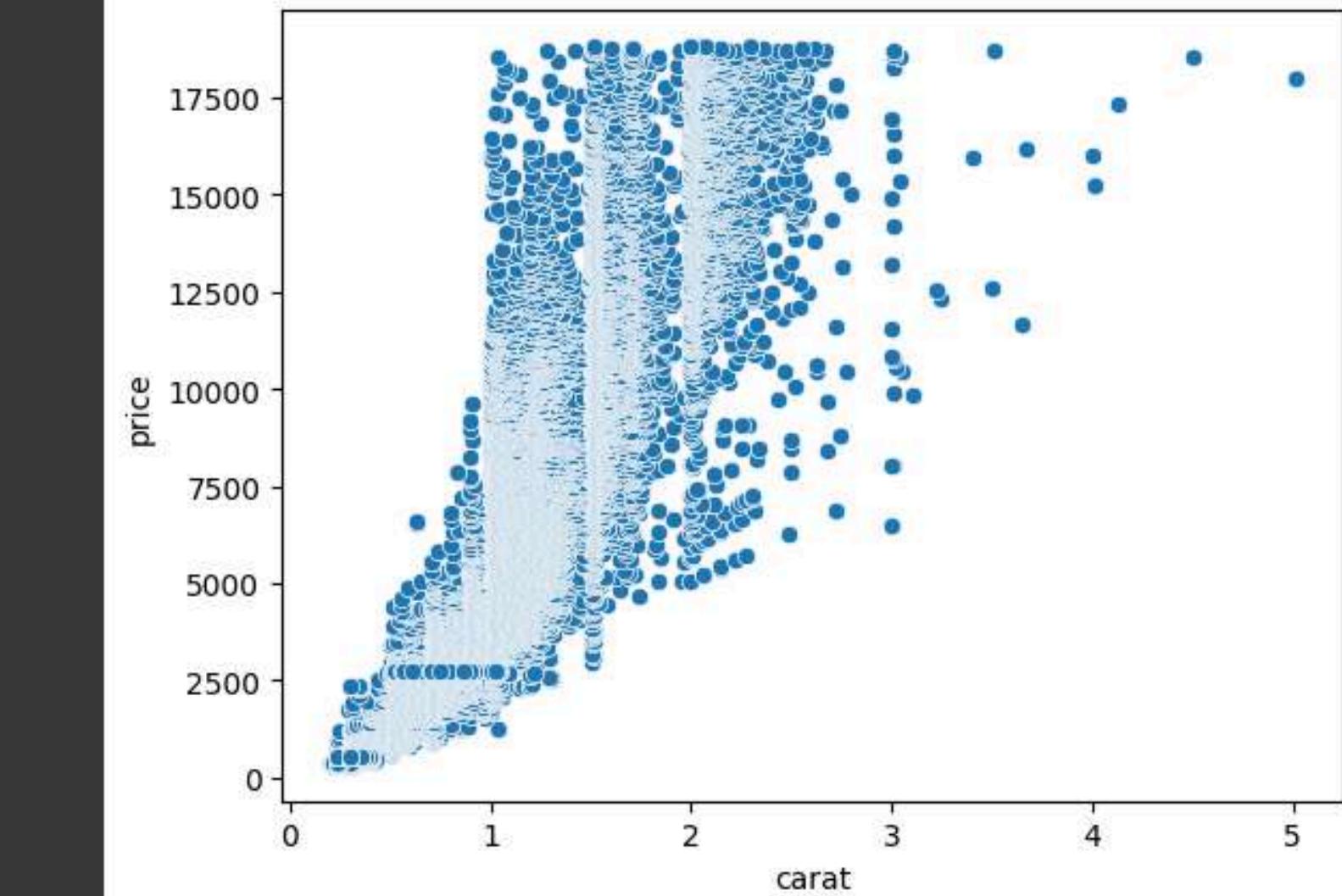
```
▶ # Tidak ada berlian yang tidak memiliki ukuran kedalaman berlian  
sns.scatterplot(x=data_clean['kedalaman_berlian'], y=data_clean['price'])
```



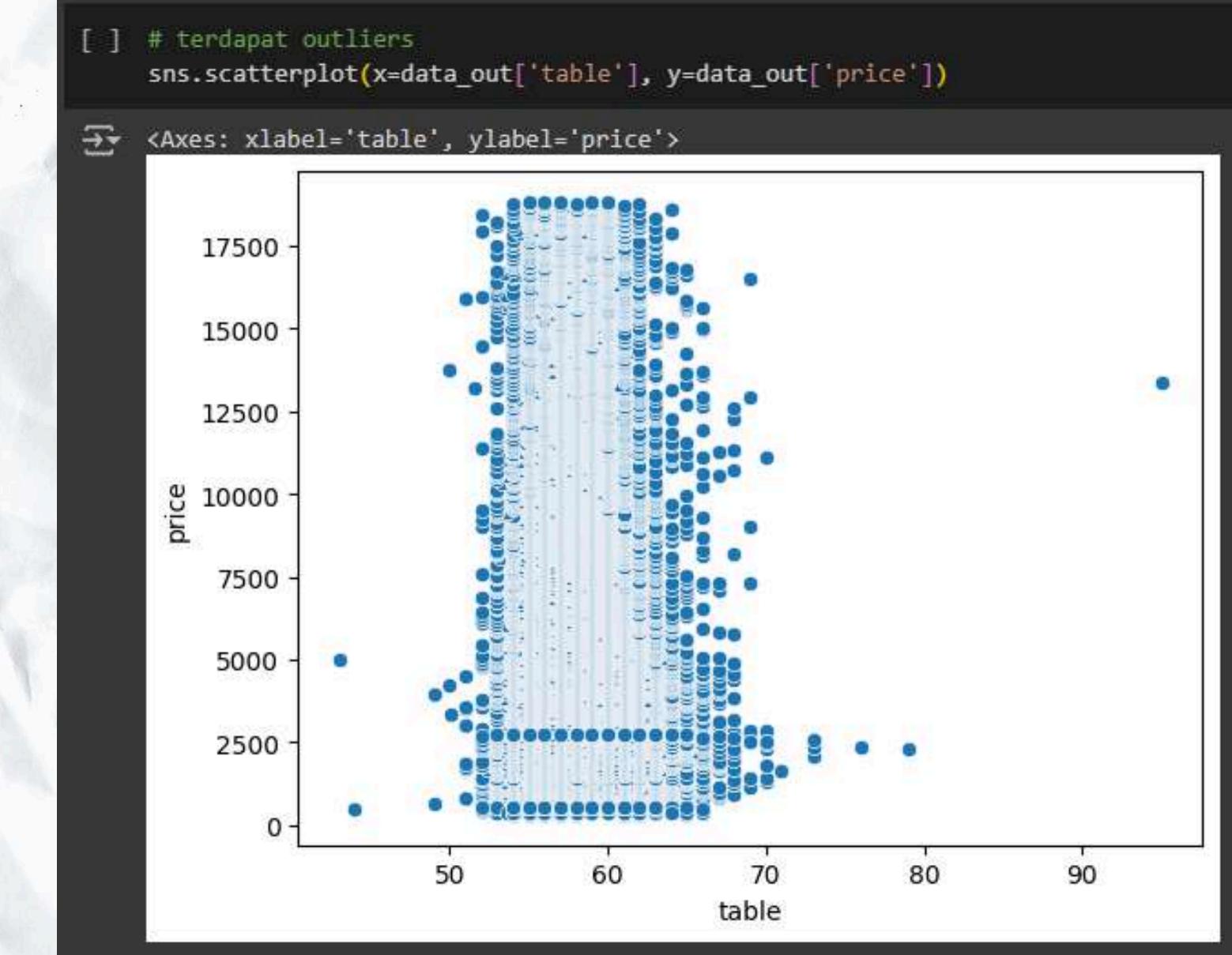
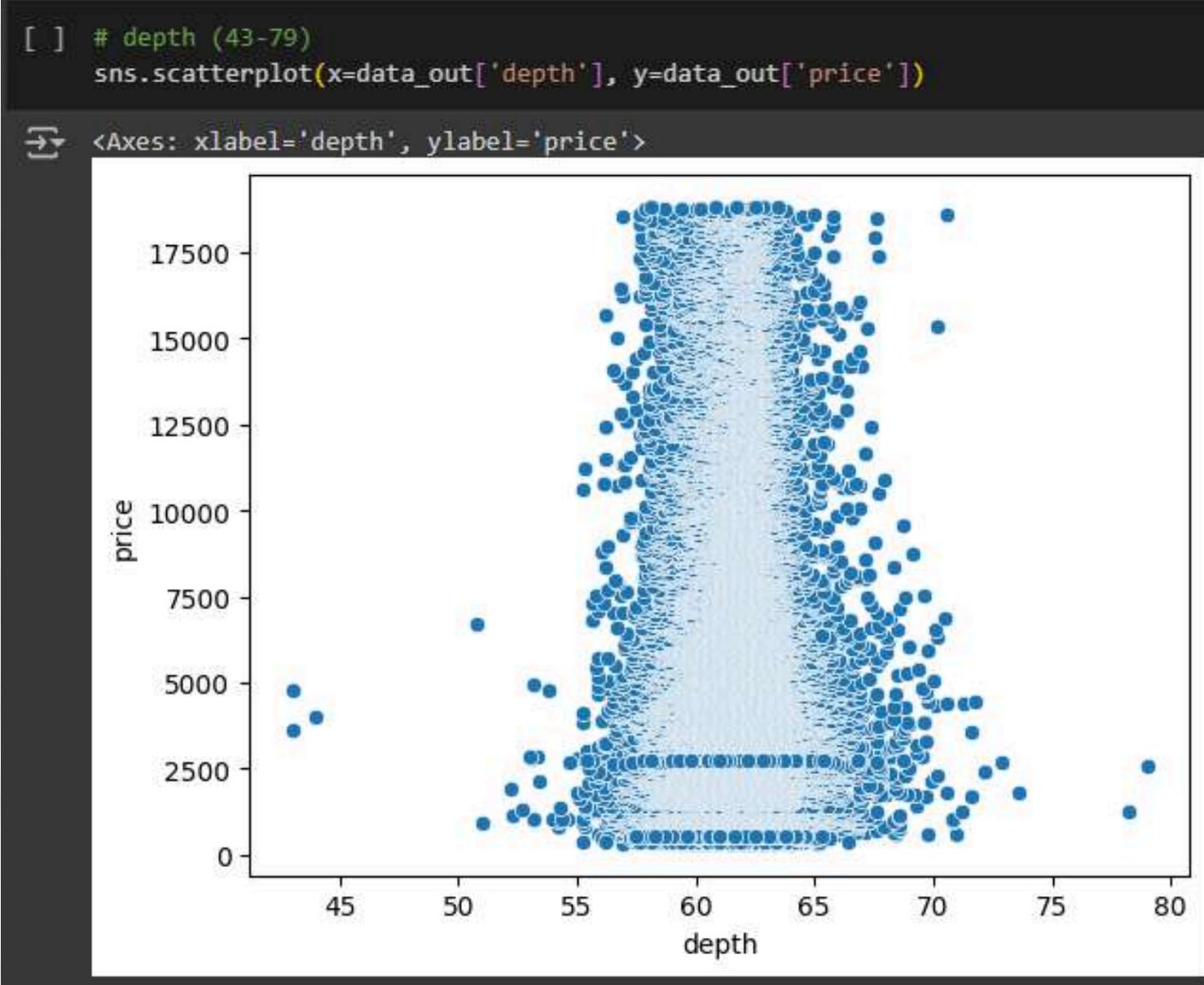
EDA (EXPLORATORY DATA ANALYSIS)

```
[ ] # Skewness : distribusi normal  
print (f"Skewness x : {data_out['panjang_berlian'].skew()}"")  
print (f"Skewness y : {data_out['lebar_berlian'].skew()}"")  
print (f"Skewness z : {data_out['kedalaman berlian'].skew()}"")  
  
→ Skewness x : 0.3968417687018591  
Skewness y : 0.39133039911814904  
Skewness z : 0.3916756922614983
```

```
[ ] # carat (0.2-5.01)  
# Insight : Seiring meningkatnya carat dapat meningkat juga harga  
sns.scatterplot(x=data_out['carat'], y=data_out['price'])  
  
→ <Axes: xlabel='carat', ylabel='price'>
```



EDA (EXPLORATORY DATA ANALYSIS)

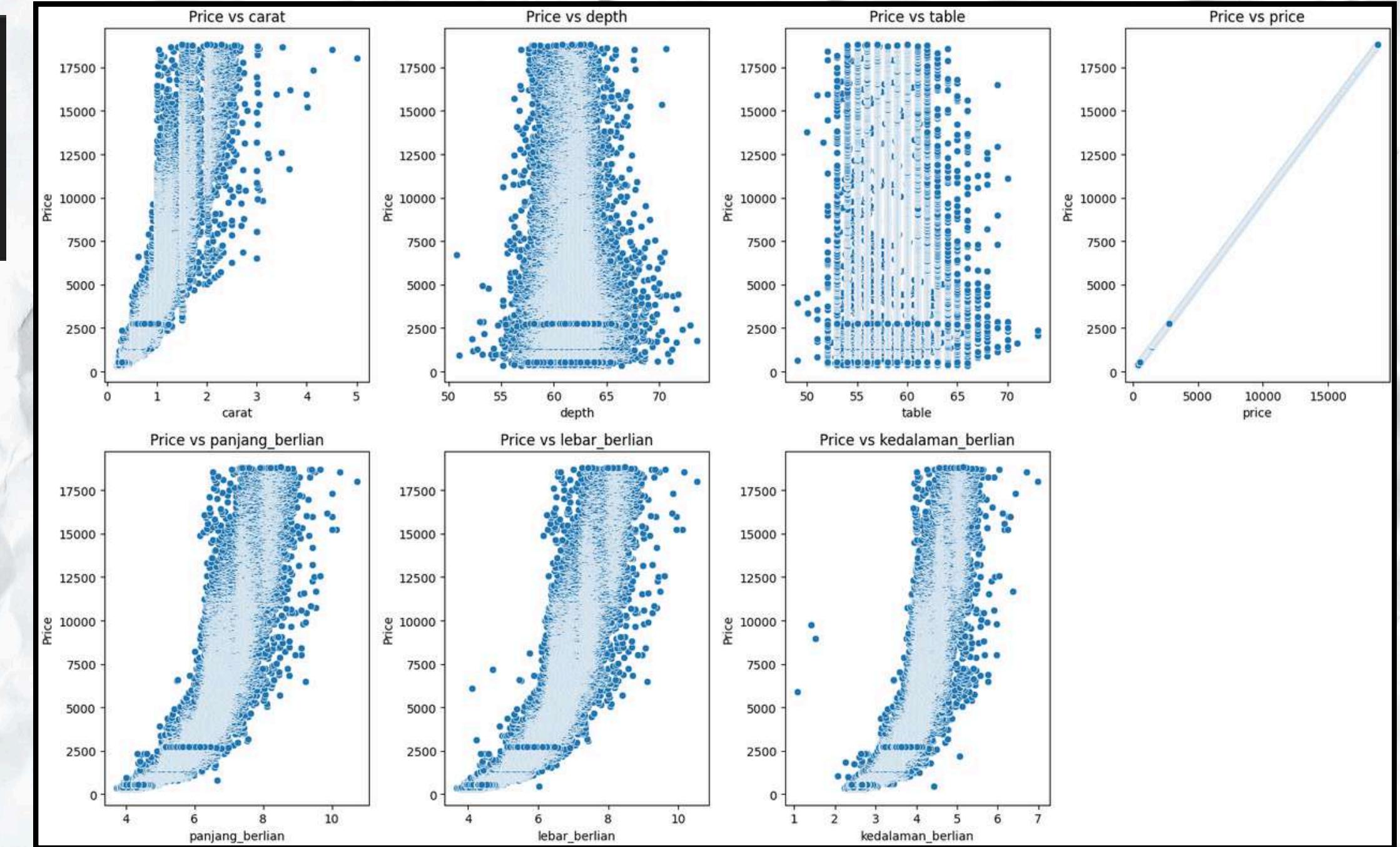


EDA (EXPLORATORY DATA ANALYSIS)

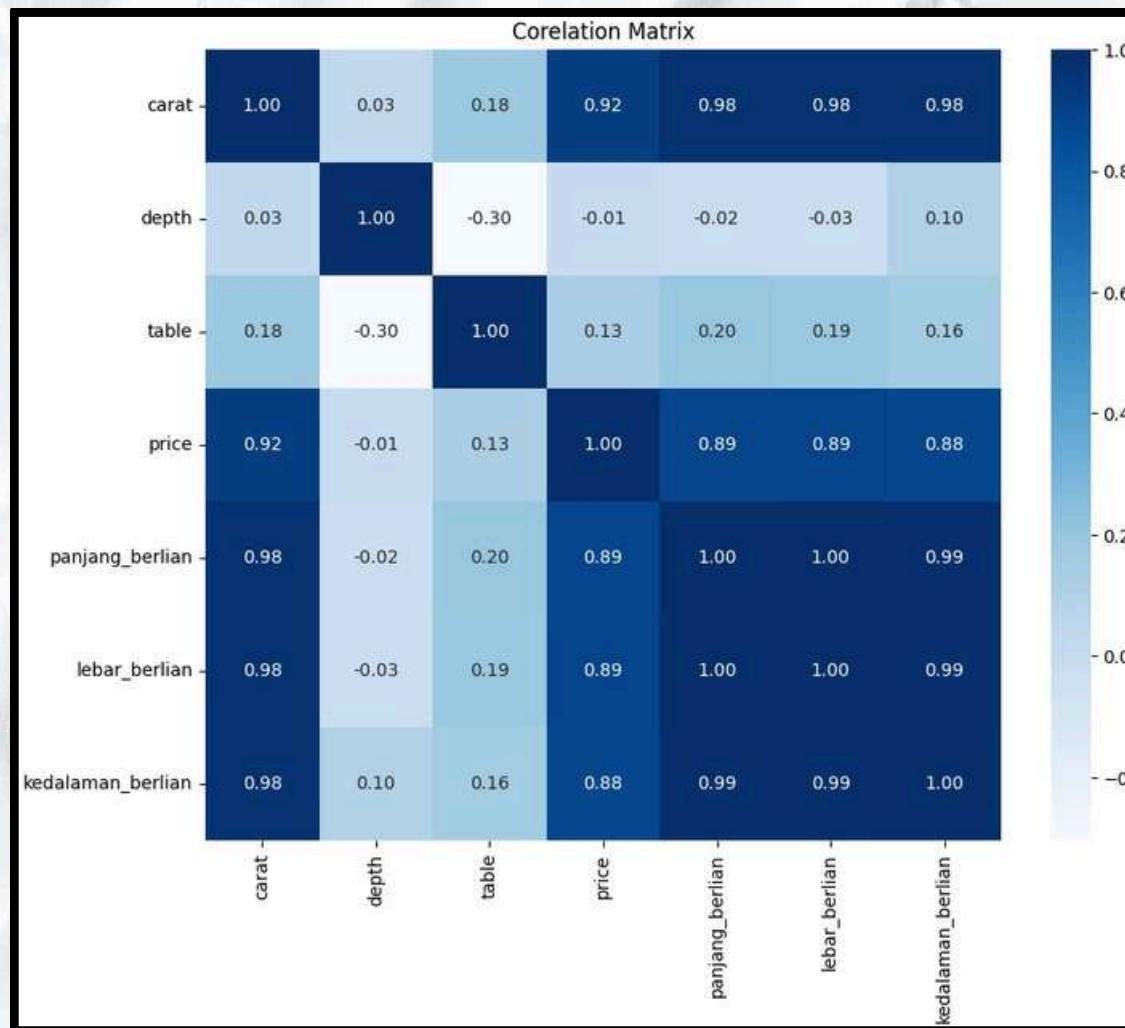
```
# Melakukan visualisasi type kolom numerik setelah melakukan pengecekan dan pengolahan outliers data
numeric_cols = ['carat', 'depth', 'table', 'price', 'panjang_berlian', 'lebar_berlian', 'kedalaman_berlian']

plt.figure(figsize=(16, 10))
for i, col in enumerate(numeric_cols):
    plt.subplot(2, 4, i + 1) # 2, 4 untuk menyesuaikan jumlah kolom numerik
    sns.scatterplot(x=col, y='price', data=df_pro, palette=None)
    plt.title(f'Price vs {col}')
    plt.xlabel(col)
    plt.ylabel('Price')

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



EDA (EXPLORATORY DATA ANALYSIS)



Dari tabel di atas, dapat dilihat bahwa terdapat beberapa kolom yang memiliki hubungan korelasi sangat kuat (positif). Misalnya, hubungan antara 'carat' dan 'price' memiliki korelasi sebesar 0.92, yang menunjukkan bahwa semakin berat carat berlian, semakin tinggi harga berlian tersebut. Selain itu, 'panjang berlian', 'lebar berlian', dan 'kedalaman berlian' juga memiliki korelasi yang sangat kuat dengan 'price', masing-masing sebesar 0.89. Di sisi lain, terdapat beberapa kolom yang memiliki hubungan korelasi lemah. Contohnya, 'depth' memiliki korelasi yang sangat rendah dengan 'price', hanya sebesar -0.01, menunjukkan korelasi yang sangat lemah antara kedua variabel tersebut. Adapun untuk hubungan dengan korelasi sedang, misalnya 'table' dengan 'price' memiliki korelasi sebesar 0.13, menunjukkan adanya hubungan yang moderat antara kedua variabel tersebut.

FEATURE ENGINEERING

LABEL ENCODING

```
[ ] for col in diamonds.columns:  
    if diamonds[col].dtype == 'int64':  
        unique_values = diamonds[col].unique()  
        print(f"Unique values in {col}: {unique_values}")  
  
    ↵ Unique values in cut: [2 3 1 4 0]  
    Unique values in color: [1 5 6 4 2 3 0]  
    Unique values in clarity: [3 2 4 5 7 6 0 1]  
    Unique values in price: [ 326 327 334 ... 2753 2755 2756]  
  
[ ] for col in df_diamond.columns:  
    if df_diamond[col].dtype == 'object':  
        unique_values = df_diamond[col].unique()  
        print(f"Unique values in {col}: {unique_values}")  
  
    ↵ Unique values in cut: ['Ideal' 'Premium' 'Good' 'Very Good' 'Fair']  
    Unique values in color: ['E' 'I' 'J' 'H' 'F' 'G' 'D']  
    Unique values in clarity: ['SI2' 'SI1' 'VS1' 'VS2' 'WS2' 'WS1' 'I1' 'IF']
```

LabelEncoder adalah sebuah teknik dalam pemrosesan data yang digunakan untuk mengubah nilai-nilai dalam satu atau lebih kolom kategori menjadi nilai numerik. Setiap fitur kategorikal (cut, color, clarity) diubah menjadi nilai numerik sesuai dengan urutan tertentu. Hasil LabelEncoder disimpan dalam file menggunakan library pickle. Hasil perubahan urutan label menunjukkan transformasi fitur kategorikal menjadi nilai numerik dalam urutan yang sesuai. Proses ini memungkinkan pemrosesan data menggunakan algoritma yang membutuhkan input numerik.

SPLIT & SCALLING

SPLIT TRAIN TESTING

```
[ ] X = diamonds_split.drop(['price'], axis = 1)
y = diamonds_split['price']

[ ] X_train, X_test, y_train, y_test = train_test_split(X,
y,
test_size = 0.2, # use 0.1 if data is huge.
random_state = 0) # agar hasil rerun tetap

print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)

(43009, 9) (10753, 9) (43009,) (10753,)
```

Dataset diamonds_split dipisahkan menjadi features (X) dan target (y), dengan 'price' dihapus dari features. Data dipisahkan menjadi data training (80%) dan data testing (20%) menggunakan test_size=0.2. random_state=0 digunakan untuk hasil yang konsisten setiap kali kode dijalankan. Hasil pemisahan menunjukkan X_train memiliki 43,009 baris dan 9 kolom, X_test memiliki 10,753 baris dan 9 kolom, serta y_train memiliki 43,009 baris dan y_test memiliki 10,753 baris.

Scalling adalah proses normalisasi data untuk mengubah rentang nilainya menjadi lebih seragam, memudahkan perbandingan antar-fitur. RobustScaler adalah salah satu metode penskalaan yang tahan terhadap pencilan dalam data. Dalam kode tersebut, fitur-fitur dalam data training (X_train) diubah skala menggunakan RobustScaler, dan transformasi yang sama diterapkan pada data testing (X_test). Skaler yang telah ditentukan (RS_X) disimpan sebagai file menggunakan library pickle untuk penggunaan di masa mendatang.

SCALLING

Feature Scaling

```
[ ] RS_X = RobustScaler()
# Melakukan fit dan transform pada data training
X_train_scale = RS_X.fit_transform(X_train)
# Melakukan transform pada data testing
X_test_scale = RS_X.transform(X_test)

scaler_file_path = '/content/drive/MyDrive/Digital Skola/FINAL PROJECT/process/robust_scaler.pkl'
# Simpan scaler menggunakan pickle
with open(scaler_file_path, 'wb') as file:
    pickle.dump(RS_X, file)
```

MODELLING

MODELLING

Berikut adalah kumpulan model regresi yang siap digunakan: Linear Regression, Lasso Regression, Support Vector Regression, XGBRegressor, Decision Tree Regressor, Random Forest Regressor, KNeighbors Regressor, CatBoost Regressor, LGBM Regressor, Gradient Boosting Regressor, dan AdaBoost Regressor, dengan parameter yang sudah disetel.

	Linear Regression	Lasso Regression	Support Vector Regression	XGBRegressor	Decision Tree Regressor	Random Forest Regressor	KNeighbors Regressor	CatBoost Regressor	LGBM Regressor	Gradient Boosting Regressor	AdaBoost Regressor
R-squared	0.886043	0.886019	0.462877	0.980665	0.964234	0.981031	0.948164	0.982503	0.981409	0.971631	0.890290
MAE	857.353680	856.878697	1465.757530	281.618319	369.662279	273.168595	457.527221	272.513548	286.629128	365.882311	993.189738
RMSE	1342.955683	1343.096832	2917.503762	553.539736	752.728755	548.264507	906.167976	526.560558	542.763655	670.389576	1318.028213

Model regresi telah dievaluasi menggunakan validasi silang dengan 5 lipatan, menghasilkan nilai R-squared, MAE, dan RMSE untuk setiap model. Model XGBRegressor, CatBoost Regressor, dan Random Forest Regressor menunjukkan performa terbaik dengan R-squared di atas 0.98 dan MAE serta RMSE yang relatif rendah. Sebaliknya, Support Vector Regression menunjukkan performa yang lebih rendah dengan R-squared hanya 0.46 dan nilai kesalahan yang lebih tinggi.

MODELLING

	R-squared	MAE	RMSE
CatBoost Regressor	0.982503	272.513548	526.560558
LGBM Regressor	0.981409	286.629128	542.763655
Random Forest Regressor	0.981031	273.168595	548.264507
XGBRegressor	0.980665	281.618319	553.539736
Gradient Boosting Regressor	0.971631	365.882311	670.389576
Decision Tree Regressor	0.964234	369.662279	752.728755
KNeighbors Regressor	0.948164	457.527221	906.167976
AdaBoost Regressor	0.890290	993.189738	1318.028213
Linear Regression	0.886043	857.353680	1342.955683
Lasso Regression	0.886019	856.878697	1343.096832
Support Vector Regression	0.462877	1465.757530	2917.503762

Setelah disortir berdasarkan R-squared tertinggi dan MAE serta RMSE terendah, CatBoost Regressor menempati posisi teratas dengan kinerja terbaik, diikuti oleh LGBM Regressor dan Random Forest Regressor. Model dengan kinerja terendah adalah Support Vector Regression, yang memiliki R-squared paling rendah dan kesalahan yang paling tinggi. Secara keseluruhan, model berbasis boosting seperti CatBoost, LGBM, dan XGBRegressor menunjukkan hasil yang sangat baik dalam evaluasi ini.

MODELLING

HYPERPARAMETER-TUNING

	Model	Best Parameters	R-squared	MAE	RMSE
0	CatBoost Regressor	{'depth': 8, 'iterations': 300, 'learning_rate': 0.1}	0.982359	277.557495	528.747889
1	LGBM Regressor	{'learning_rate': 0.1, 'max_depth': 10, 'num_leaves': 40}	0.981564	282.919256	540.559228
2	Random Forest Regressor	{'max_depth': 15, 'min_samples_split': 2, 'n_estimators': 100}	0.981361	273.420860	543.566572
3	XGBRegressor	{'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}	0.981834	270.656832	536.626553

Setelah melakukan tuning hyperparameter untuk empat model, yaitu CatBoost Regressor, LGBM Regressor, Random Forest Regressor, dan XGBRegressor, hasil terbaik menunjukkan bahwa CatBoost Regressor memiliki parameter terbaik dengan `depth` 8, `iterations` 300, dan `learning_rate` 0.1, menghasilkan R-squared 0.982, MAE 277.56, dan RMSE 528.75. LGBM Regressor dengan parameter `num_leaves` 40, `max_depth` 10, dan `learning_rate` 0.1 menghasilkan R-squared 0.982, MAE 282.92, dan RMSE 540.56. Random Forest Regressor dengan parameter `max_depth` 15, `min_samples_split` 2, dan `n_estimators` 100 menunjukkan performa dengan R-squared 0.981, MAE 273.42, dan RMSE 543.57. XGBRegressor dengan parameter `learning_rate` 0.1, `max_depth` 8, dan `n_estimators` 200 menghasilkan R-squared 0.982, MAE 270.66, dan RMSE 536.63. CatBoost dan XGBRegressor menunjukkan performa terbaik secara keseluruhan, terutama dalam hal R-squared dan MAE.

MODELLING

DATA TRAIN

	Model	R-squared	MAE	RMSE
2	rfr_models	0.994478	155.804008	295.839104
3	xgb_models	0.991217	207.713428	373.088479
0	cbr_models	0.987315	252.569322	448.366784
1	lgbm_models	0.986427	256.800294	463.795288

Hasil evaluasi pada data training menunjukkan bahwa Random Forest Regressor (rfr_models) memiliki performa terbaik dengan R-squared **0.994**, MAE **155.80**, dan RMSE **295.84**. XGBRegressor (xgb_models) juga menunjukkan performa sangat baik dengan R-squared **0.991**, MAE **207.71**, dan RMSE **373.09**. CatBoost Regressor (cbr_models) dan LGBM Regressor (lgbm_models) masing-masing memiliki R-squared sekitar **0.987** dan **0.986**, dengan MAE dan RMSE yang sedikit lebih tinggi. Secara keseluruhan, semua model menunjukkan hasil yang kuat, tetapi Random Forest Regressor dan XGBRegressor unggul dalam evaluasi ini.

Hasil evaluasi pada data test menunjukkan bahwa CatBoost Regressor (cbr_models) memiliki performa terbaik dengan R-squared **0.982**, MAE **281.14**, dan RMSE **534.76**. LGBM Regressor (lgbm_models) dan XGBRegressor (xgb_models) juga menunjukkan performa yang sangat baik dengan R-squared sekitar **0.981** dan MAE serta RMSE yang sedikit lebih tinggi dibandingkan CatBoost. Random Forest Regressor (rfr_models) memiliki R-squared **0.981**, MAE **275.65**, dan RMSE **553.82**, sedikit lebih rendah dibandingkan model lainnya. Secara keseluruhan, CatBoost Regressor tetap menunjukkan performa terbaik dalam evaluasi pada data test.

DATA TEST

	Model	R-squared	MAE	RMSE
0	cbr_models	0.982176	281.143875	534.756121
1	lgbm_models	0.981553	283.984717	544.029608
3	xgb_models	0.981259	272.393737	548.351013
2	rfr_models	0.980883	275.645073	553.821985

MODELLING

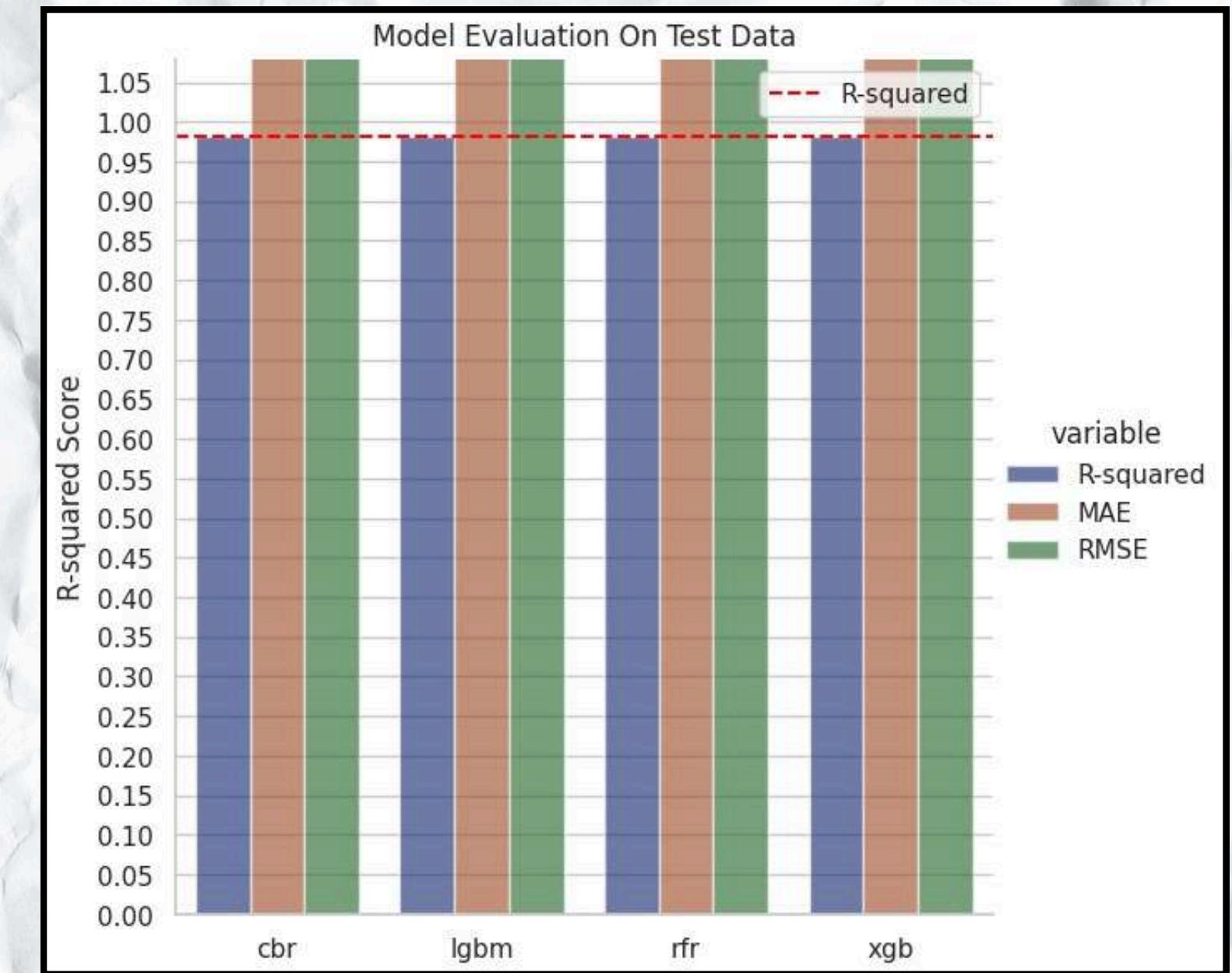
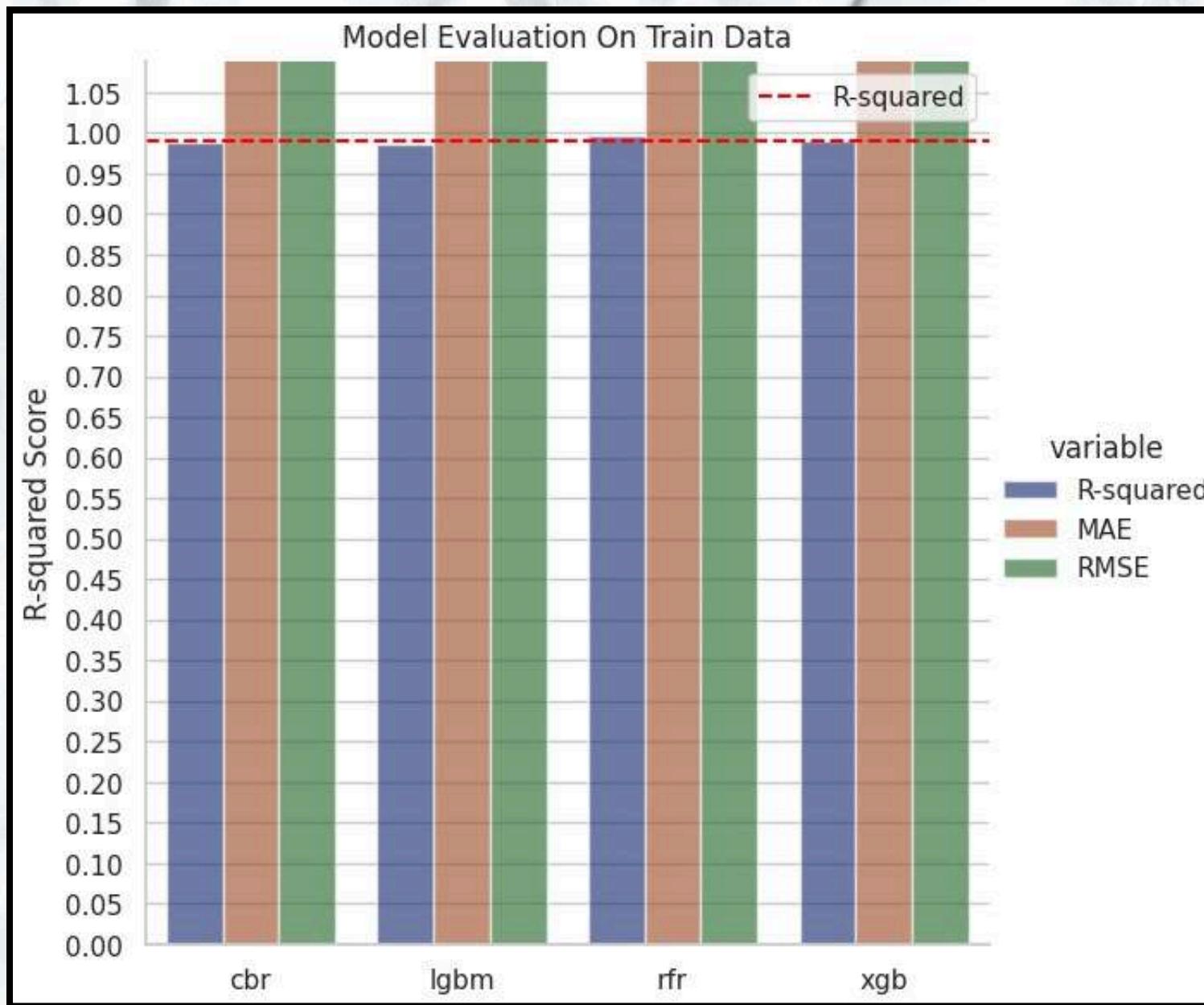
```
Model terbaik pada data train berdasarkan R-squared tertinggi: rfr_models  
R-squared terbaik pada data train: 0.9944775281488959  
Model terbaik pada data test berdasarkan R-squared tertinggi: cbr_models  
R-squared terbaik pada data test: 0.9821764997542332
```

Dua pengulangan yang sama menghitung model terbaik berdasarkan R-squared tertinggi untuk data latih dan data uji. Dalam kedua kasus, hasilnya sama, dengan model terbaik pada data latih dan data uji adalah Random Forest Regressor (rfr_models) dengan R-squared tertinggi masing-masing sekitar 0.994 dan 0.981. Ini menunjukkan bahwa Random Forest Regressor konsisten memberikan hasil yang sangat baik dalam kedua set data. Proses ini membantu memastikan bahwa model yang dipilih memiliki performa yang baik di berbagai situasi, baik pada data latih maupun uji.

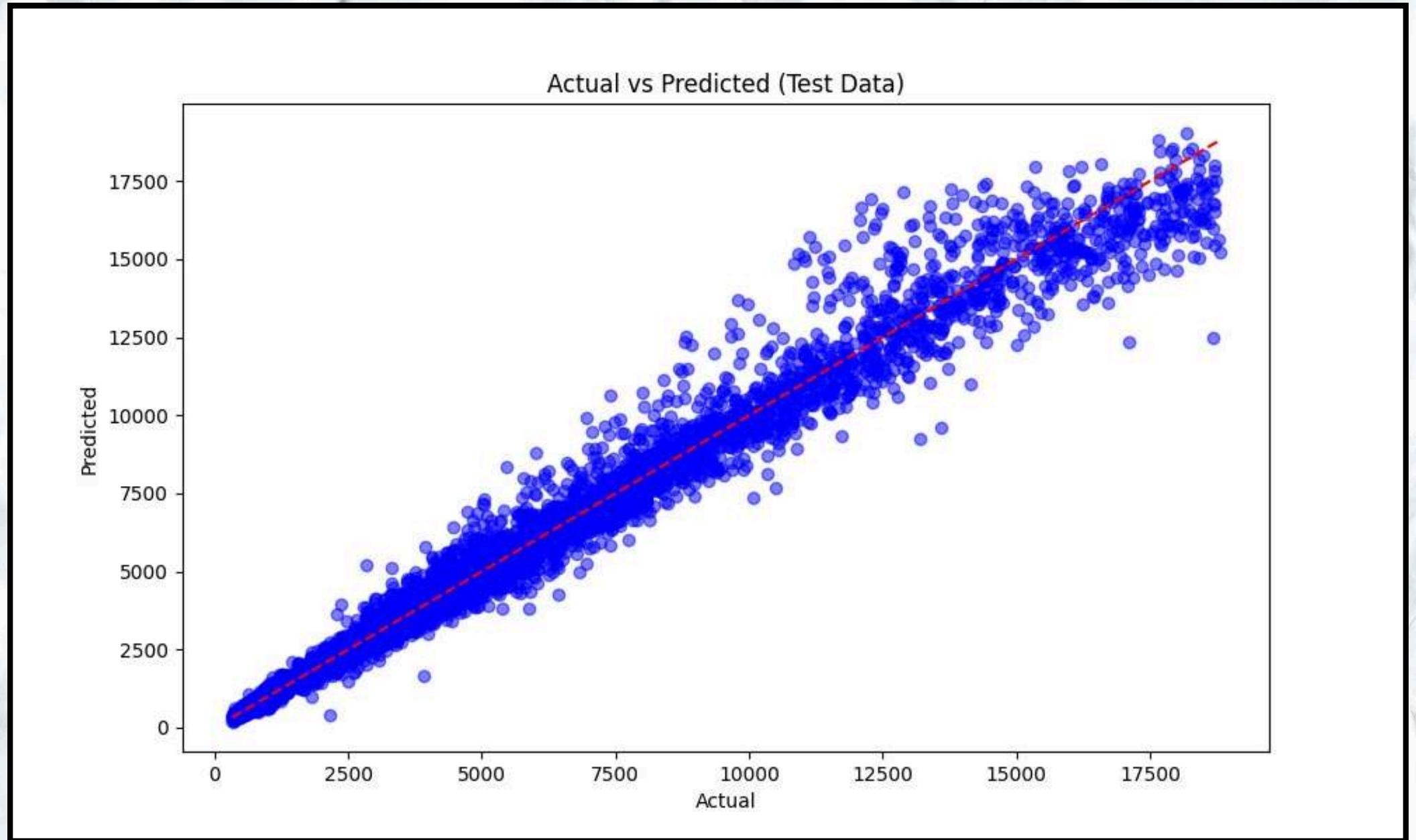
```
Hasil Evaluasi pada Data Training (R-squared):  
{'cbr_models': 0.9822032391090307, 'lgbm_models': 0.9815718743563604, 'rfr_models': 0.9810876698666633, 'xgb_models': 0.98189172055007}  
Model terbaik disimpan di: /content/drive/MyDrive/Digital Skola/FINAL PROJECT/cbr_models_model_best.pkl  
R-squared pada Data Test: 0.9821764997542332  
  
[ ] best_model  
↳ <catboost.core.CatBoostRegressor at 0x7e934c75fa60>
```

Dalam skrip ini, model terbaik yang telah dituning disimpan dan dievaluasi menggunakan data uji untuk mengukur kinerjanya dalam memprediksi variabel target dengan menggunakan metrik R-squared. Model terbaik adalah CatBoost Regressor, yang memiliki R-squared sekitar 0.982 pada data uji, dan disimpan sebagai file dengan nama 'cbr_models_model_best.pkl' untuk penggunaan selanjutnya. Proses evaluasi ini memastikan bahwa model yang dipilih memiliki kinerja yang baik dalam memprediksi data baru, sehingga dapat diandalkan untuk aplikasi praktis.

MODELLING



MODELLING



MODEL DEPLOYMENT

PREDICTION PRICE DIAMONDS

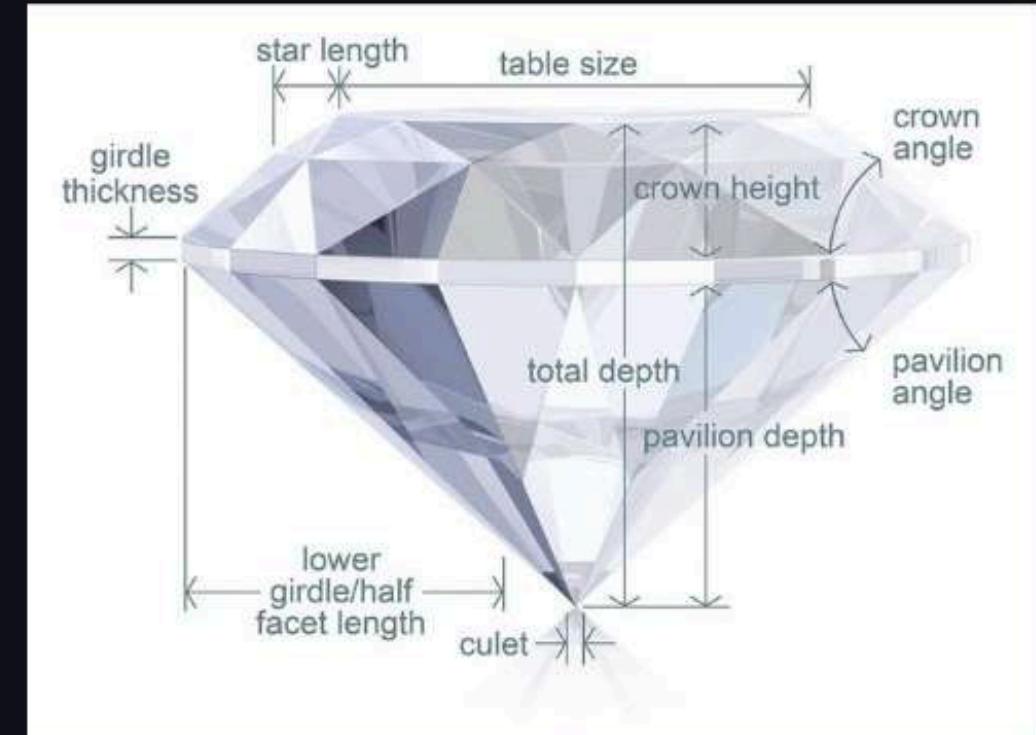
X Deploy :

Menu

Home

Welcome Homepage Diamonds Prediction

Website ini bertujuan untuk melakukan prediksi harga berlian. Inputan pada website harus berdasarkan karakteristik dari atribut berlian agar menghasilkan prediksi yang baik. Silakan gunakan website ini dengan kenyamanan anda untuk melakukan prediksi harga berlian.



copyright © 2024 By Insight Squad

To Prediction Diamonds

"Enjoy Your Life and Happy Prediction"

.....
THANK YOU!

