LINEAR REGRESSION

UNTUK MEMPREDIKSI HARGA MOBIL BEKAS

Data Mining

Presented By :
Shifa Fanisatuz Zahra

LANGKAH PENGOLAHAN

Pre-Processing 01

Profiling Data
EDA
Data Cleaning
Data Transformations
Data Final

Modelling 02

Memisahkan fitur & target
Membagi data training & testing
Data scalling
Model Linear Regression

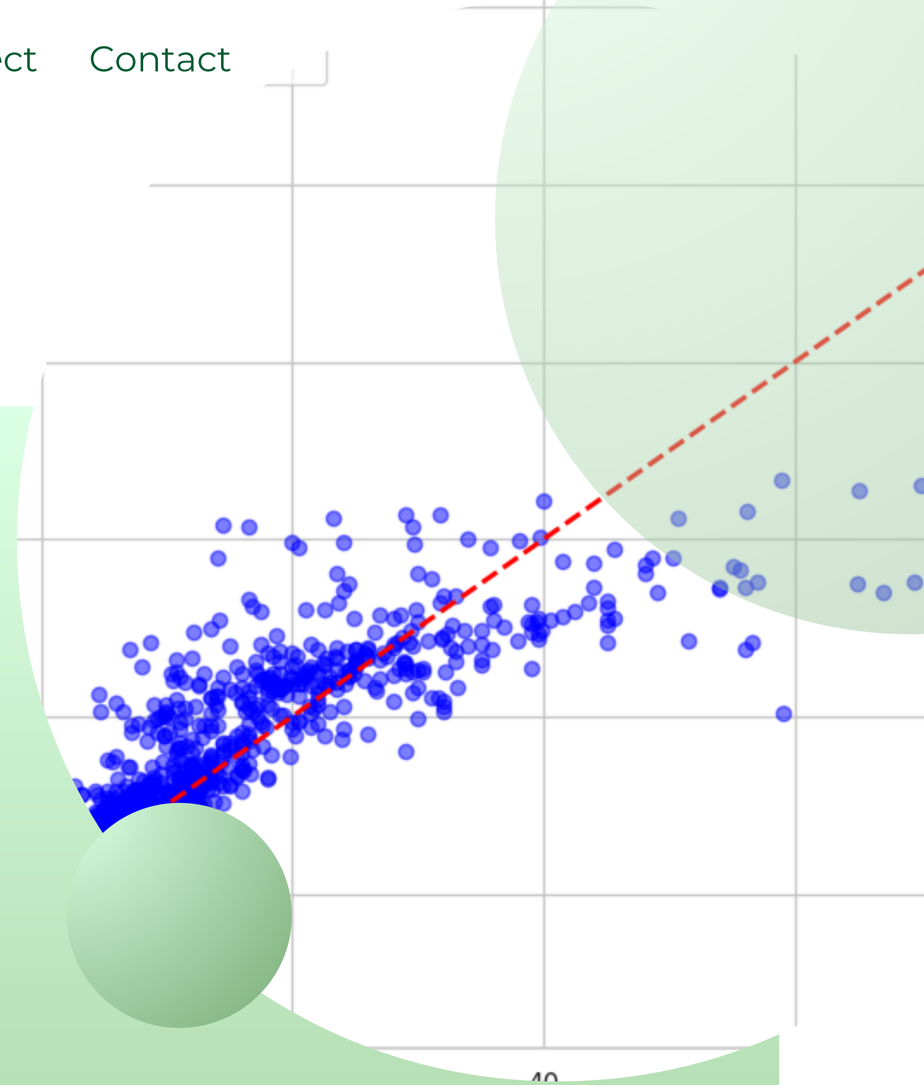
Evaluasi Model 03

R^2
MAE
MSE
RMSE

Presented By :
Shifa Fanisatuz Zahra

Apa itu Linear Regression

INTRODUCTION



Linear Regression

Metode analisis yang digunakan untuk memprediksi nilai suatu variabel (target) berdasarkan hubungan linear dengan satu atau lebih variabel lain (prediktor). Tujuannya untuk menemukan garis terbaik yang menggambarkan pola hubungan dalam data

Tujuan

Memprediksi harga mobil bekas berdasarkan data yang ada

DATA PROFILING

Sumber Dataset (Github)

```
# load data Used Cars Price Prediction
!wget https://raw.githubusercontent.com/FarrelllAdityaaa/dataset-uts-datamining/refs/heads/main/used_cars_price_fiks.csv
```

Target

Price

Kolom Numerik

8 kolom

Kolom Kategori

5 kolom

```
# Menampilkan informasi dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6019 entries, 0 to 6018
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Unnamed: 0         6019 non-null   int64
 1   Name               6019 non-null   object
 2   Location           6019 non-null   object
 3   Year               6019 non-null   int64
 4   Kilometers_Driven  5719 non-null   float64
 5   Fuel_Type          6019 non-null   object
 6   Transmission       6019 non-null   object
 7   Owner_Type         6019 non-null   object
 8   Mileage            6017 non-null   float64
 9   Engine             5983 non-null   float64
 10  Power              5876 non-null   float64
 11  Seats              5977 non-null   float64
 12  Price              6019 non-null   float64
dtypes: float64(6), int64(2), object(5)
memory usage: 611.4+ KB
```

EDA

(Exploratory Data Analysis)

```
# Membuuat kolom brand dari Name
df['Brand'] = df['Name'].str.split().str[0]

# Cek jumlah brand unik
print(df['Brand'].nunique())
print(df['Brand'].value_counts().head(10))
```

```
31
Brand
Maruti           1211
Hyundai          1107
Honda             608
Toyota            411
Mercedes-Benz     318
Volkswagen        315
Ford              300
Mahindra          272
BMW               267
Audi              236
Name: count, dtype: int64
```

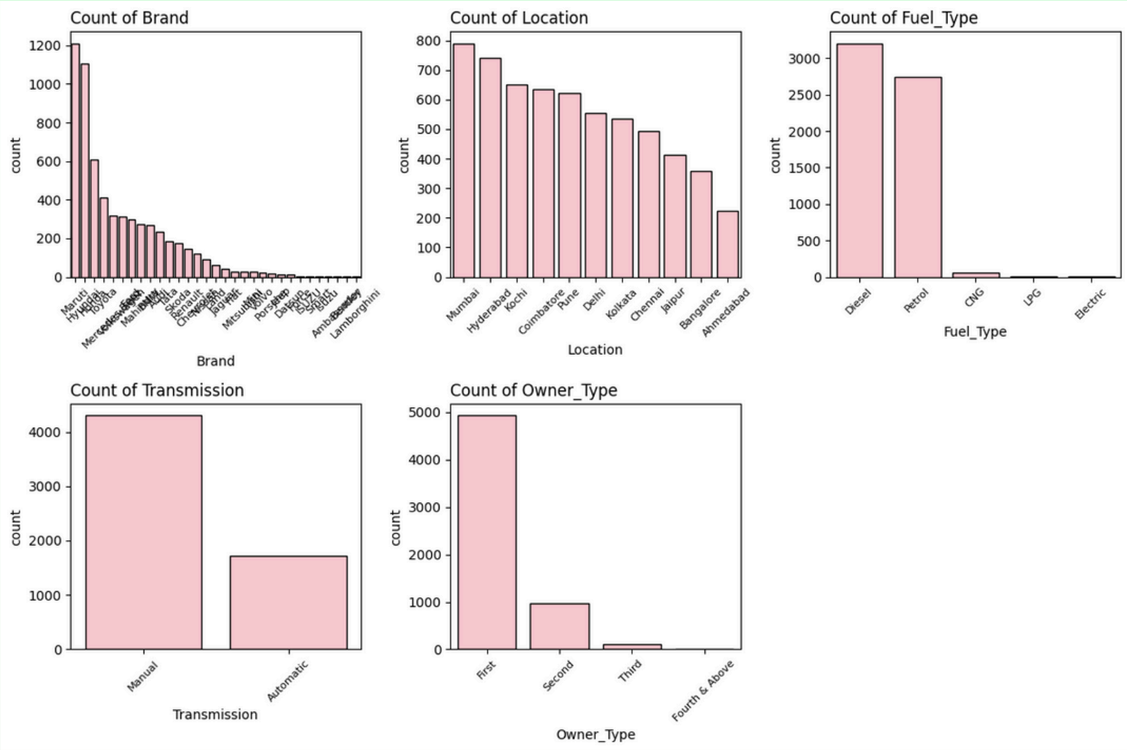
Mengubah kolom Name jadi Brand karena di kolom tersebut sebenarnya yang lebih penting diambil adalah mereknya, bukan detail nama mobil secara lengkap

EDA

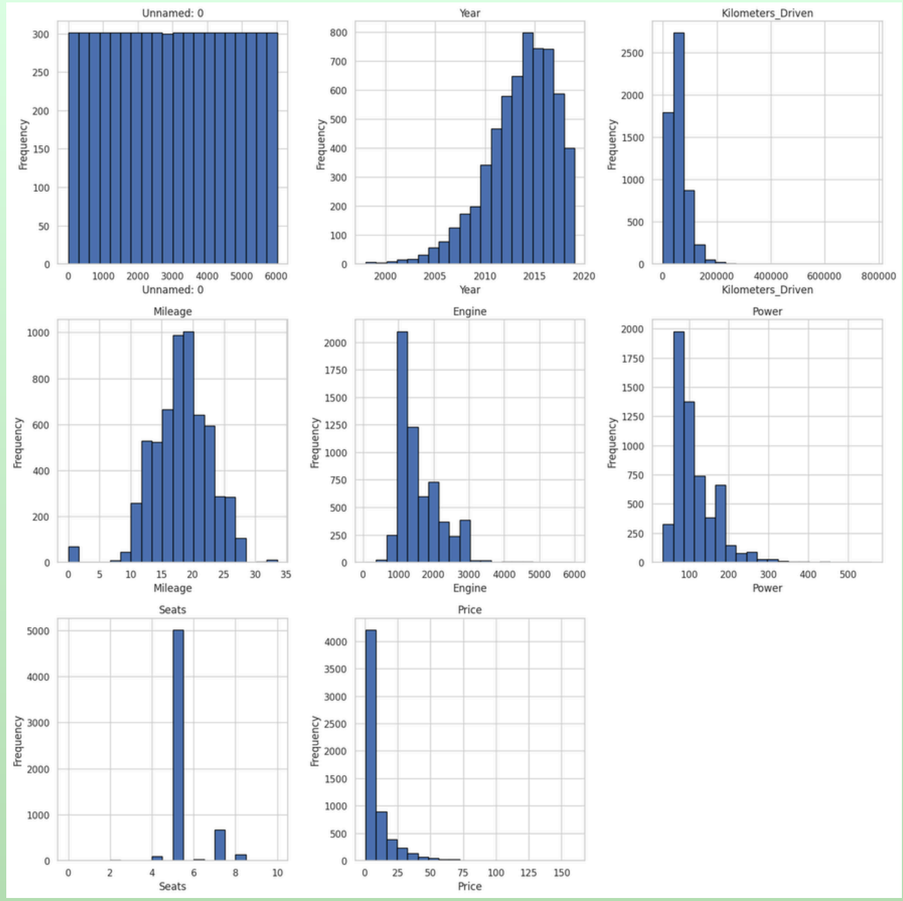
(Exploratory Data Analysis)

DISTRIBUSI KATEGORIKAL & NUMERIK

Kategorikal



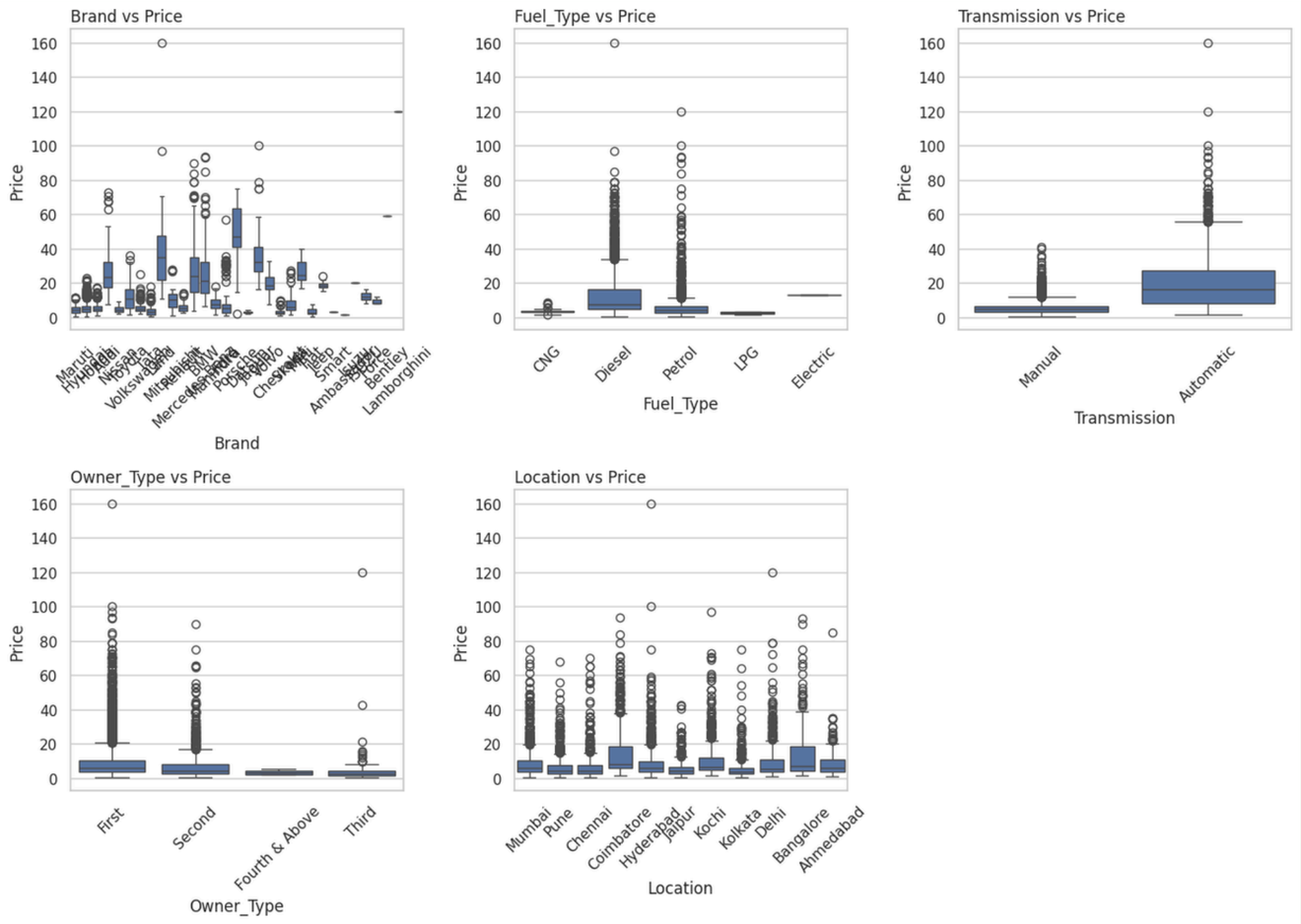
Numerik



EDA

(Exploratory Data Analysis)

HUBUNGAN KATEGORI DENGAN PRICE

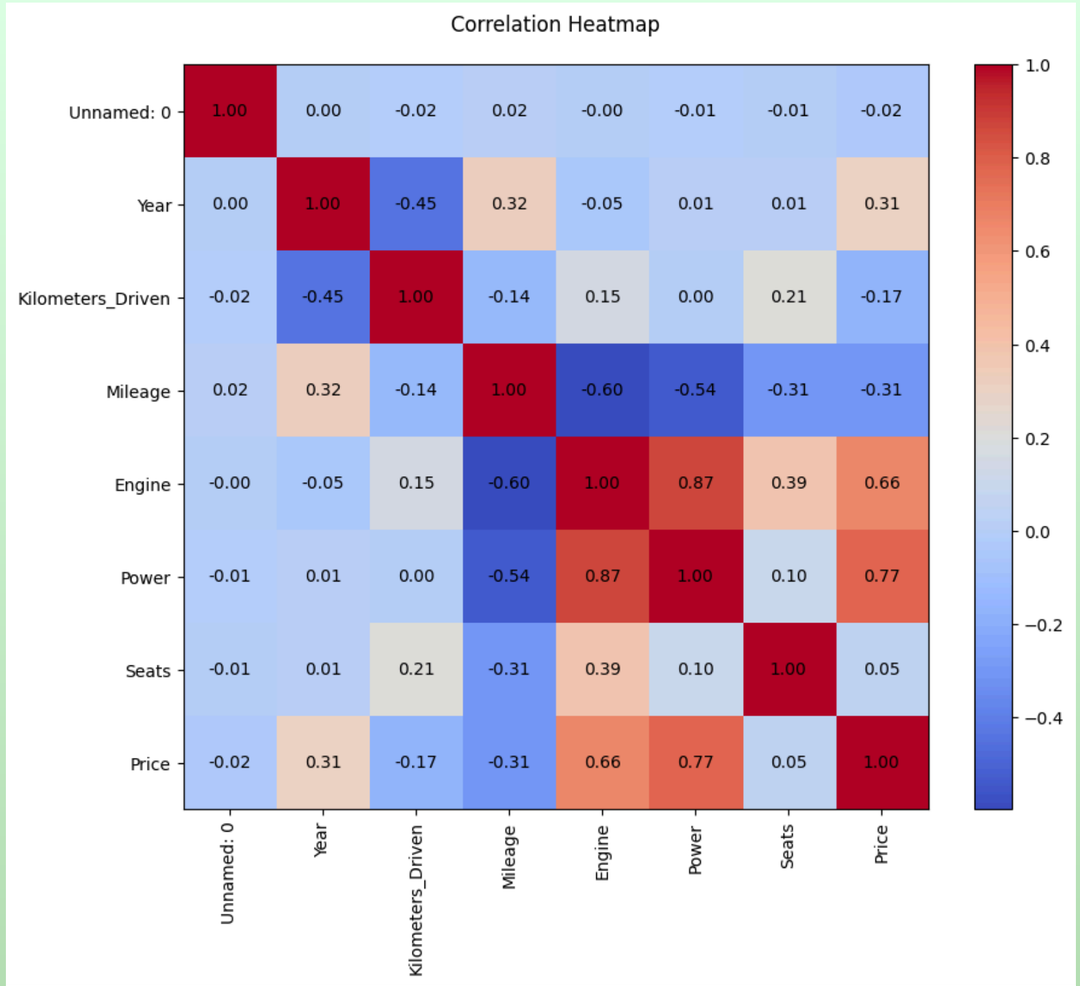


- Brand: Setiap merek memiliki rentang harga yang berbeda, menunjukkan bahwa brand berpengaruh signifikan terhadap harga mobil.
- Fuel Type: Mobil dengan bahan bakar diesel umumnya lebih mahal dibandingkan yang lainnya, menandakan jenis bahan bakar turut memengaruhi harga.
- Transmission: Mobil transmisi otomatis cenderung lebih mahal daripada manual.
- Owner Type: Mobil dengan pemilik pertama (First Owner) memiliki harga lebih tinggi, karena semakin sering berpindah tangan, harga biasanya turun.
- Location: Harga mobil berbeda antar kota, di mana kota besar cenderung memiliki harga lebih tinggi dibanding kota kecil

EDA

(Exploratory Data Analysis)

KORELASI ANTAR DATA NUMERIK



Dari hasil heatmap, terlihat bahwa harga mobil (Price) dipengaruhi oleh beberapa variabel, terutama Power, Engine, dan Year. Semakin besar tenaga, kapasitas mesin, dan tahun keluaran mobil, harga cenderung lebih tinggi. Sementara itu, Kilometers_Driven berhubungan negatif dengan harga—semakin jauh jarak tempuh, harga makin rendah. Variabel Seats hampir tidak berpengaruh, sedangkan Mileage berpengaruh kecil. Selain itu, Engine dan Power juga tampak memiliki hubungan yang kuat satu sama lain

DATA CLEANING

Remove Null Value

	0
Unnamed: 0	0
Name	0
Location	0
Year	0
Kilometers_Driven	300
Fuel_Type	0
Transmission	0
Owner_Type	0
Mileage	2
Engine	36
Power	143
Seats	42
Price	0
Brand	0

→

```
Missing value setelah cleaning:

Unnamed: 0           0
Name                 0
Location             0
Year                 0
Kilometers_Driven    0
Fuel_Type            0
Transmission         0
Owner_Type           0
Mileage              0
Engine               0
Power                0
Seats                0
Price                0
Brand                0
dtype: int64
```

karena kalau digabung nilai null ini bakal cukup banyak, jadi saya menanganinya dengan beberapa cara, yaitu:

- imputasi median untuk kolom Kilometers_Driven, Engine, dan Power
- imputasi modus untuk kolom seats
- dan menghapus pada kolom mileage

Source Code:

```
# Mengatasi dengan imputasi median untuk kolom Kilometers_Driven, Engine, dan Power
median_cols = ['Kilometers_Driven', 'Engine', 'Power']
for col in median_cols:
    df_clean[col] = df_clean[col].fillna(df_clean[col].median())
```

```
# Mengatasi dengan imputasi mean untuk kolom Seats
df_clean['Seats'] = df_clean['Seats'].fillna(df_clean['Seats'].mode()[0])
```

```
# Mengatasi dengan menghapus, karena cuman 2
df_clean = df_clean.dropna(subset=['Mileage'])
```

DATA CLEANING

Remove Duplicate Value & Remove Column

Tidak ada duplikat data di dataset ini

```
# Cek duplikasi dataset
df_clean.duplicated().sum()
```

```
np.int64(0)
```

- kolom Unnamed dia tidak punya arti apa-apa, cuman index bawaan dari dataset
- Kolom Name sudah diganti Brand

Source Code:

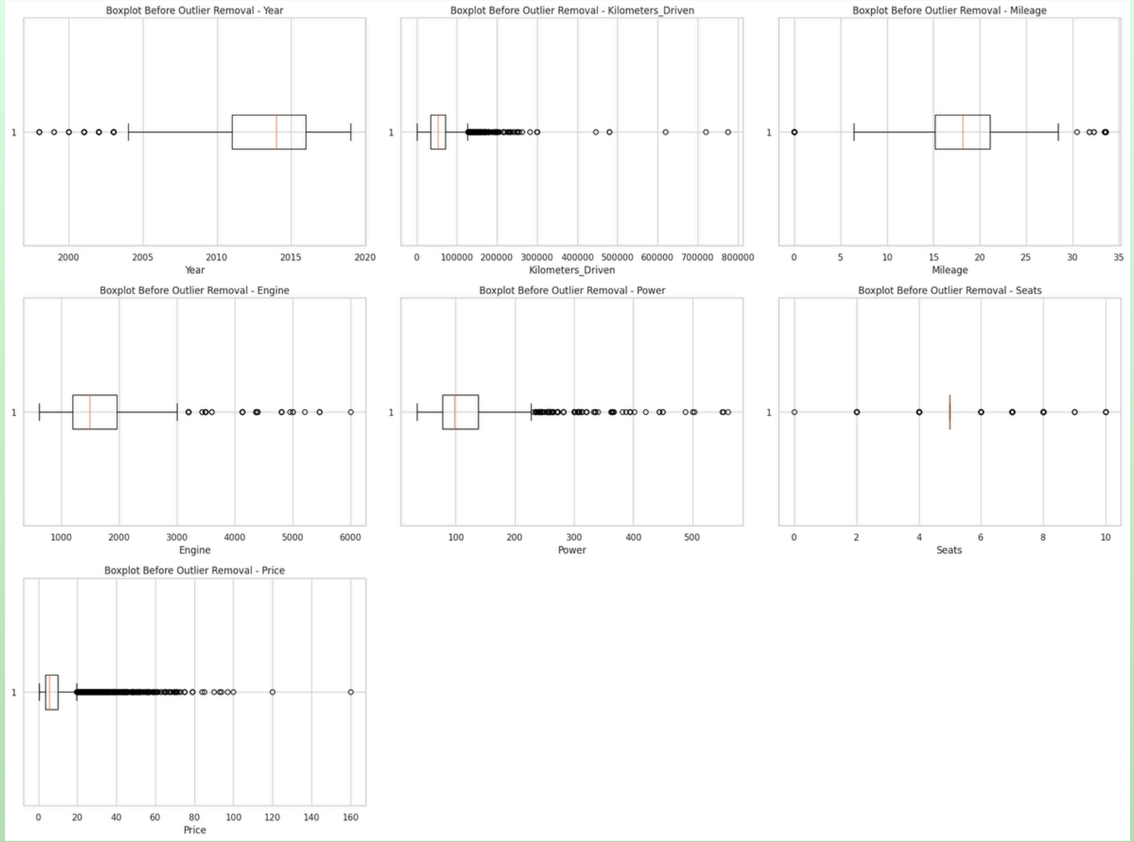
```
# Menghapus kolom Unnamed dan Name
df_clean.drop(['Unnamed: 0', 'Name'], axis=1, inplace=True)

print(df_clean.columns)
```

```
Index(['Location', 'Year', 'Kilometers_Driven', 'Fuel_Type', 'Transmission',
       'Owner_Type', 'Mileage', 'Engine', 'Power', 'Seats', 'Price', 'Brand'],
      dtype='object')
```

DATA CLEANING

Remove Outliers

Mengecek boxplot sebelum outliers diatasi



Jumlah outlier per kolom (IQR method):

	Outlier Count
Year	50
Kilometers_Driven	224
Mileage	82
Engine	61
Power	239
Seats	963
Price	718

DATA CLEANING

Menangani Outliers

Jumlah outlier beda-beda tiap kolom. Outlier ini perlu dicek lagi apakah masuk akal atau justru error data, supaya bisa diputuskan cara menanganinya

- untuk Year, saya batasi antara 1990–2022 karena di luar itu sudah terlalu lama atau terlalu baru
- untuk Seats, saya hapus data di luar 2–9 kursi. Alasannya saya tidak menganggap kursi lebih dari 9 wajar karena kalau dilihat dari brand yang ada di dataset ini, semuanya seperti mobil pribadi, bukan bus atau kendaraan besar
- untuk kolom numerik lain seperti Kilometers_Driven, Engine, Power, Mileage, dan Price, pakai capping

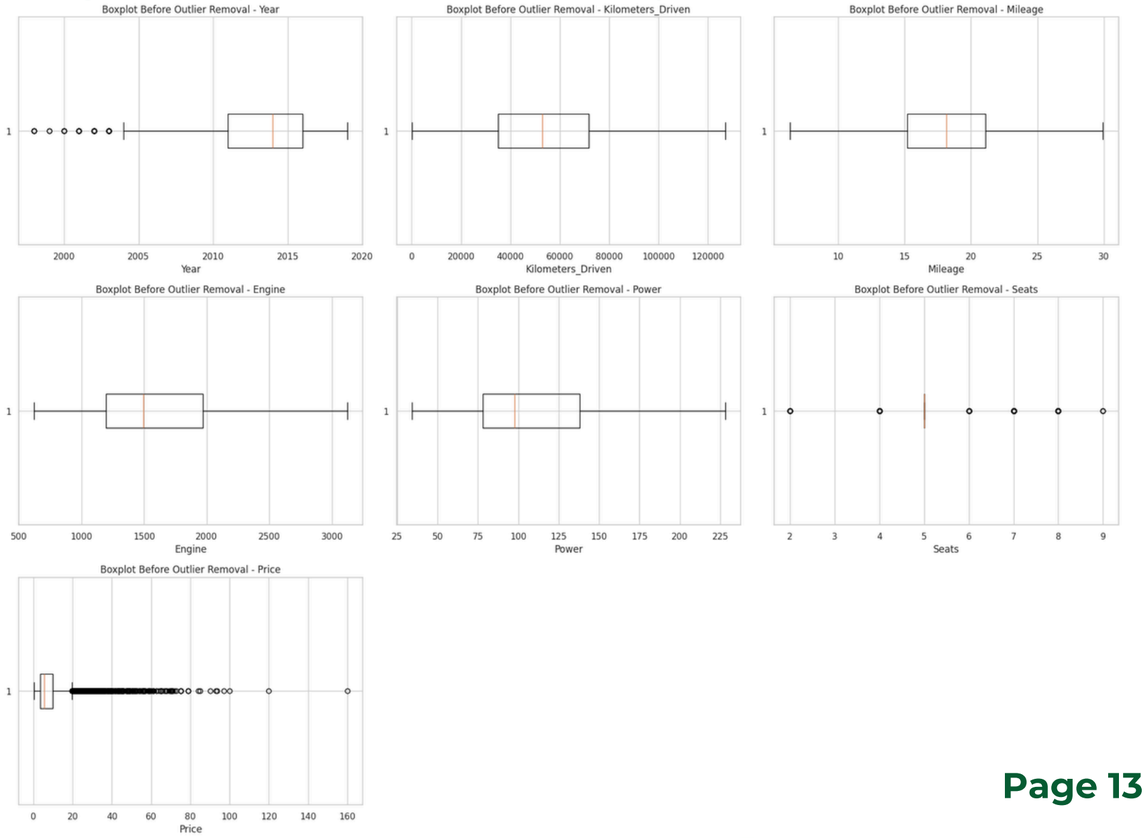
```
# Menangani outlier di kolom Year
df_outlier = df_outlier[(df_outlier['Year'] >= 1990) & (df_outlier['Year'] <= 2022)].copy()

# Menangani outlier di kolom Seats
df_outlier = df_outlier[(df_outlier['Seats'] >= 2) & (df_outlier['Seats'] <= 9)].copy()

# Menangani outlier di kolom lainnya
def cap_outliers(df, col):
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5*IQR
    upper = Q3 + 1.5*IQR

    df.loc[df[col] < lower, col] = lower
    df.loc[df[col] > upper, col] = upper

    return df

for col in ['Kilometers_Driven', 'Mileage', 'Engine', 'Power']:
    df_outlier = cap_outliers(df_outlier, col)

print("Jumlah baris setelah outlier handling:", df_outlier.shape[0])
print("\nDistribusi Seats setelah filter:\n", df_outlier['Seats'].value_counts())
print("\nOutlier sudah ditangani dengan capping & filtering")
```

DATA CLEANING

Hasil setelah diatasi Outliers

```
Jumlah baris setelah outlier handling: 6011

Distribusi Seats setelah filter:
 Seats
5.0    5054
7.0     674
8.0     134
4.0      99
6.0      31
2.0      16
9.0       3
Name: count, dtype: int64
```

Mengecek boxplot setelah diatasi outliernya



DATA CLEANING

Data Transformations

Feature Engineering

```
# Membuat kolom Age (usia mobil)
df_transformed['Age'] = 2025 - df_transformed['Year']

# Menghapus kolom Year karena sudah diganti dengan Age
df_transformed.drop('Year', axis=1, inplace=True)
```

Menambahkan kolom baru Age dari kolom Year karena umur mobil lebih relevan untuk melihat hubungannya dengan Price dibandingkan cuman menggunakan tahunnya doang secara langsung

One-Hot Encoding untuk Kategori

```
# One-Hot Encoding untuk kategori
categorical_cols = ['Location', 'Fuel_Type', 'Transmission', 'Owner_Type', 'Brand']
df_transformed = pd.get_dummies(df_transformed, columns=categorical_cols, drop_first=True)
```

One-hot encoding dilakukan pada fitur kategorikal karena model linear regression hanya bisa memproses data numerik. Dengan one-hot encoding, setiap kategori diubah menjadi kolom biner (0 dan 1) sehingga model dapat memahami pengaruh masing-masing kategori terhadap harga mobil bekas tanpa menganggap adanya urutan atau nilai numerik di antara kategori tersebut

DATA FINAL

```
print("Dimensi akhir dataset:", df_final.shape)
```

```
Dimensi akhir dataset: (6011, 54)
```

Setelah proses pre-processing, jumlah data berubah karena beberapa langkah pembersihan dan transformasi dilakukan.

Data awal memiliki 6019 baris dan 13 kolom, namun setelah:

- Menghapus data kosong, beberapa baris hilang jadi 6011 baris.
- Melakukan one-hot encoding pada fitur kategorikal, setiap kategori diubah menjadi kolom biner, jumlah kolom bertambah dari 13 menjadi 54.

Jadi, perubahan ukuran data disebabkan oleh pembersihan data dan transformasi fitur kategorikal menjadi bentuk numerik.

MODELLING

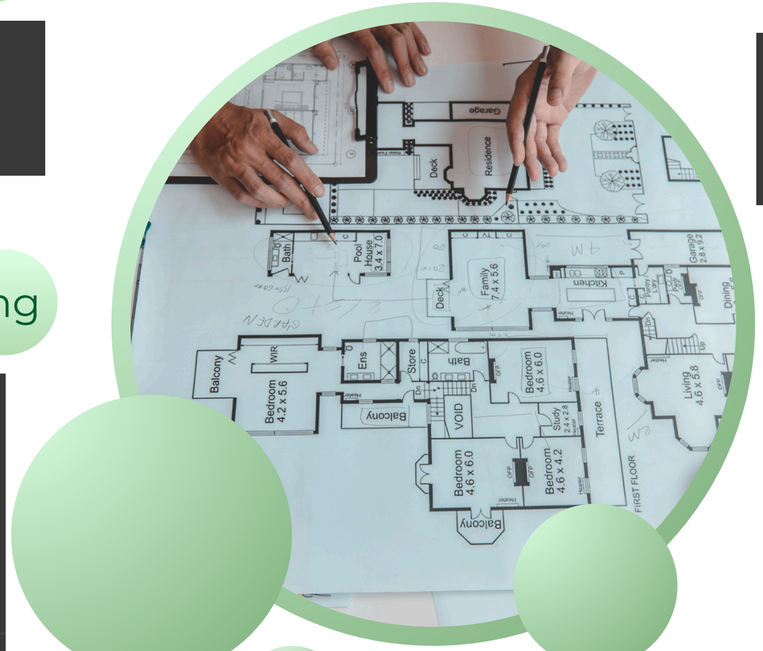
01. Memisahkan Fitur & Target

```
# Memisahkan X (fitur) dan y (target)
X = df_final.drop(columns=['Price'])
y = df_final['Price']
```

02. Membagi data Training & Testing

```
# Membagi data training (80%) & data testing(20%)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

print("Ukuran data train:", X_train.shape)
print("Ukuran data test:", X_test.shape)
```

```
Ukuran data train: (4808, 53)
Ukuran data test: (1203, 53)
```



03. Data Scalling

```
# Data Scalling
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

04. Linear Regression

```
# Tahap linear regressionnya
model = LinearRegression()
model.fit(X_train_scaled, y_train)
```

```
LinearRegression
LinearRegression()
```

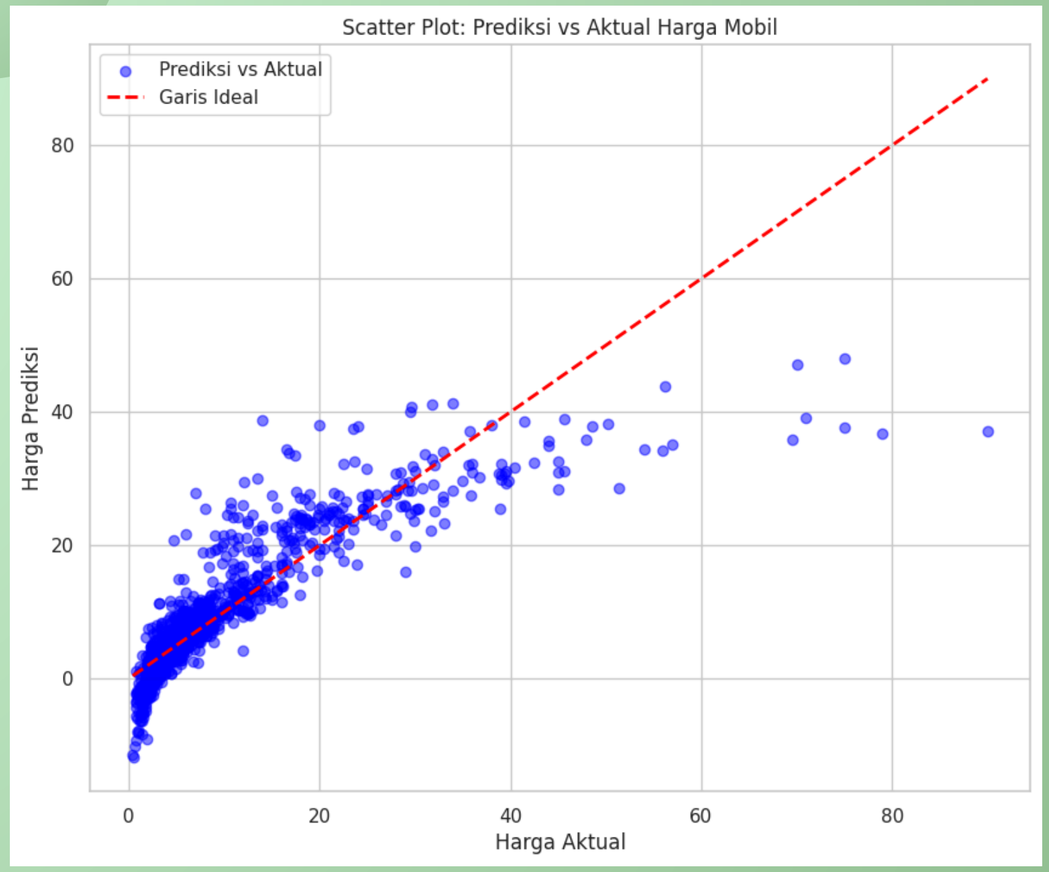
EVALUASI MODEL

```
# Evaluasi
r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print("R²   :", r2)
print("MAE  :", mae)
print("MSE  :", mse)
print("RMSE :", rmse)
```

```
R²   : 0.753632194966861
MAE  : 2.9763533831684303
MSE  : 26.277938604894207
RMSE : 5.126201186540985
```

- R^2 = 0.75 → Model mampu menjelaskan sekitar 75% variasi harga mobil, sehingga performanya bisa dianggap cukup baik.
- MAE = 2.97 → Rata-rata selisih prediksi dengan harga asli sekitar 3 satuan.
- MSE = 26.27 → Menunjukkan besarnya error kuadrat, sehingga nilainya terlihat lebih besar.
- RMSE = 5.12 → Secara rata-rata, model salah memprediksi sekitar 5 satuan harga.

VISUALISASI DENGAN SCATTER PLOT



KESIMPULAN

Model linear regression mampu memprediksi harga mobil bekas dengan cukup baik, ditunjukkan oleh nilai R^2 sebesar 0.75 yang berarti model dapat menjelaskan sebagian besar variasi harga. Nilai MAE, MSE, dan RMSE juga menunjukkan bahwa tingkat kesalahan prediksi masih tergolong wajar, sehingga model ini cukup akurat dan layak digunakan untuk analisis atau prediksi awal harga mobil bekas

THANK YOU

THANK YOU

https://colab.research.google.com/drive/1Sc-auMeguRI6TpKthXSQGlcQj3Zd2Wsq?usp=sharing

Presented By :
Shifa Fanisatuz Zahra