Visualisasi dan Analisis Data Sistem *Pricing* dan *Grading* Mobil pada Perusahaan Carro

Oleh Sumarni

Ringkasan

Visualisasi dan analisis data pada sistem pricing dan grading mobil dilakukan mengg unakan pemograman bahasa python. Data mobil sejumlah 2213 buah. Merek mobil sebelu m ditambah data baru meliputi Daihatsu, Nissan, Suzuki, Mitstubisi, Chevrolet, Kia, Isuzu dan Nissan. Kemudian ditambahkan merek mobil Hyundai, Dodge Journey, Cherry QQ. Visualisasi data dilakukan menggunakan seaborn boxplot. Akan tetapi masih terl alu general sehingga dilakukan pengembangan sistem pricing dan grading menggunakan machine learning. Machine learning digunakan untuk mempermudah dalam memprediksi h arga berdasarkan data yang telah ada. Model yang digunakan adalah Decision Tree Re gressor. Data train sejumlah 1549 dan data test sejumlah 664. Setelah dilakukan dat a testing diperoleh nilai F1-Score 64%. Akurasi dapat ditingkatkan dengan menambah jumlah data. Rekomendasi di masa depan dapat menggunakan model deep learning untuk sistem pricing dan grading mobil.

Import Libraries

`Import libaries yang akan digunakan berupa drive, pandas, numpy, matplotlib, scikit-sklearn.`

In [0]:

```
from google.colab import drive
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import datasets
from sklearn.metrics import mean_squared_error

from sklearn.tree import DecisionTreeRegressor
```

Load Data

`Load dataset dilakukan menggunakan drive.mount dari google drive dan pandas`

In [4]:

```
#Dataset yang digunakan di import dari google drive

drive.mount('/content/gdrive')
```

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

In [5]:

```
#Dataset yang diimport berupa file excel
import pandas as pd
df=pd.read_excel('gdrive/My Drive/Colab Notebooks/Dataset_Challenge_Carro_rev.xlsx')
```

```
df.head() #Mencetak 5 dataset teratas
```

Out[5]:

	TAHUN	TOP STATUS	MODEL	TRANSMISI	TYPE	HARGA AWAL	PENYESUAIAN HARGA	HARGA PANDUAN	HARGA TERENDAH	HARGA TERTINGGI	
0	2019.0	FAST	XENIA	AUTOMATIC	1.3 R BENSIN	124200000.0	0.1	138000000.0	140500000.0	182500000.0	13
1	2019.0	FAST	XENIA	MANUAL	1.3 R BENSIN	125100000.0	0.1	139000000.0	133000000.0	173000000.0	14
2	2019.0	FAST	XENIA	AUTOMATIC	1.3 R DELUXE BENSIN	126000000.0	0.1	140000000.0	148500000.0	192500000.0	14
3	2019.0	FAST	XENIA	MANUAL	1.3 R DELUXE BENSIN	129600000.0	0.1	144000000.0	141500000.0	183500000.0	14
4	2019.0	FAST	XENIA	AUTOMATIC	1.3 R SPORTY BENSIN	134100000.0	0.1	149000000.0	147500000.0	191500000.0	15

Pre-Processing Data

Pre-processing data sangat penting dilakukan untuk mengecek kelengkapan data dan memilih data yang akan dianalisis selanjutnya.

In [6]:

```
#Menampilkan dataset yang kosong
#False menunjukkan datanya lengkap
#True menunjukkan datanya ada yang kosong
df.isnull().any()
```

Out[6]:

```
TAHUN                          True
TOP STATUS                     True
MODEL                          True
TRANSMISI                      True
TYPE                           True
HARGA AWAL                     True
PENYESUAIAN HARGA              True
HARGA PANDUAN                  True
HARGA TERENDAH                 True
HARGA TERTINGGI                True
OOM                            True
POOL LEADER                    True
TOP FIX                        True
CARSOME                        True
OTO                            True
BMG tinggi\n15% - 18% - 20%    True
Selisih                        True
MEREK                          True
dtype: bool
```

In [7]:

```
#Menampilkan features dataset
df.columns
```

Out[7]:

```
Index(['TAHUN', 'TOP STATUS', 'MODEL', 'TRANSMISI', 'TYPE', 'HARGA AWAL',
       'PENYESUAIAN HARGA', 'HARGA PANDUAN', 'HARGA TERENDAH',
       'HARGA TERTINGGI', 'OOM', 'POOL LEADER', 'TOP FIX', 'CARSOME', 'OTO'
```

```
       'BMG tinggi\n15% - 18% - 20%', 'Selisih', 'MEREK'],
      dtype='object')
```

In [8]:

```
#Menyeleksi data yang akan digunakan
#Pada pandas drop digunakan untuk menghapus label apabila axis = 1 maka yang dihapus baris, axis = 0 yang dihapus kolom
#Label yang digunakan untuk analisis data berupa tahun, model, transmisi, dan harga panduan
df.drop(['TOP STATUS', 'TYPE', 'HARGA AWAL','PENYESUAIAN HARGA', 'HARGA TERENDAH','HARGA TERTINGGI', 'OOM', 'POOL LEADER', 'TOP FIX', 'CARSOME', 'OTO','BMG tinggi\n15% - 18% - 20%', 'Selisih'], axis=1, inplace=True)
df.head()
```

Out[8]:

	TAHUN	MODEL	TRANSMISI	HARGA PANDUAN	MEREK
0	2019.0	XENIA	AUTOMATIC	138000000.0	DAIHATSU
1	2019.0	XENIA	MANUAL	139000000.0	DAIHATSU
2	2019.0	XENIA	AUTOMATIC	140000000.0	DAIHATSU
3	2019.0	XENIA	MANUAL	144000000.0	DAIHATSU
4	2019.0	XENIA	AUTOMATIC	149000000.0	DAIHATSU

In [9]:

```
#Mengisi data yang kosong dengan menggunakan rata-rata dari dataset terutama untuk mengisi data harga panduan yang masih kosong
mobil_df=df.fillna(df.mean())
mobil_df.head()
```

Out[9]:

	TAHUN	MODEL	TRANSMISI	HARGA PANDUAN	MEREK
0	2019.0	XENIA	AUTOMATIC	138000000.0	DAIHATSU
1	2019.0	XENIA	MANUAL	139000000.0	DAIHATSU
2	2019.0	XENIA	AUTOMATIC	140000000.0	DAIHATSU
3	2019.0	XENIA	MANUAL	144000000.0	DAIHATSU
4	2019.0	XENIA	AUTOMATIC	149000000.0	DAIHATSU

In [10]:

```
#Mengecek kelengkapan data
#Hasilnya False semua menunjukkan data telah lengkap
mobil_df.isnull().any()
```

Out[10]:

```
TAHUN            False
MODEL             True
TRANSMISI         True
HARGA PANDUAN    False
MEREK             True
dtype: bool
```

In [11]:

```
mobil_df = mobil_df.dropna(axis=0)
mobil_df.head()
```

Out[11]:

	TAHUN	MODEL	TRANSMISI	HARGA PANDUAN	MEREK

	TAHUN	MODEL	TRANSMISI	HARGA PANDUAN	MEREK
0	2019.0	XENIA	AUTOMATIC	138000000.0	DAIHATSU
1	2019.0	XENIA	MANUAL	139000000.0	DAIHATSU
2	2019.0	XENIA	AUTOMATIC	140000000.0	DAIHATSU
3	2019.0	XENIA	MANUAL	144000000.0	DAIHATSU
4	2019.0	XENIA	AUTOMATIC	149000000.0	DAIHATSU

In [12]:

```
#Jumlah Dataset Mobil : 2213 mobil
# 5 Label
mobil_df.shape
```

Out[12]:

```
(2213, 5)
```

In [13]:

```
#Menampilkan index merek mobil
mobil_df.loc[:,"MEREK"].value_counts()
```

Out[13]:

```
DAIHATSU          738
SUZUKI            524
NISSAN            470
MITSUBISHI        154
ISUZU             104
CHEVROLET          75
KIA                66
DATSUN             46
HYUNDAI            31
DODGE JOURNEY       3
CHERY QQ            2
Name: MEREK, dtype: int64
```

In [14]:

```
#Menampilkan model mobil
mobil_df.loc[:,"MODEL"].value_counts().index
```

Out[14]:

```
Index(['XENIA', 'GRAND LIVINA', 'X-TRAIL', 'GRAN MAX', 'APV', 'TERIOS',
       'PANTHER', 'SIRION', 'AYLA', 'SERENA', 'PAJERO SPORT', 'ERTIGA',
       'PICANTO', 'KARIMUN', 'SWIFT', 'CARRY', 'LUXIO', 'MARCH', 'SPLASH',
       'CAPTIVA', 'EVALIA', 'PAJERO', 'KARIMUN WAGON R', 'SIGRA',
       'GRAND VITARA', 'GO+', 'JUKE', 'L300', 'X-OVER', 'GO', 'XPANDER',
       'IGNIS', 'SPIN', 'HYUNDAI ASCENT VERNA', 'KUDA', 'BALENO',
       'CHEVROLET CAPTIVA', 'ARENA', 'CHEVROLET NEW CAPTIVA',
       'CHEVROLET ZAFIRA', 'APV ARENA', 'NEO BALENO', 'WAGON R', 'TARUNA',
       'HYUNDAI AVEGA', 'KARIMUN WAGON', 'HYUNDAI MATRIX', 'BALENO NEXT-G',
       'CHEVROLET ALL NEW AVEO', 'DODGE JOURNEY', 'GO PANCA', 'APV LUXURY',
       'HYUNDAI GRAND AVEGA', 'CHERY QQ', 'GO+ PANCA', 'KARIMUN WAGON R GS',
       'HYUNDAI I24', 'HYUNDAI I21', 'HYUNDAI I25', 'HYUNDAI I23',
       'HYUNDAI I20', 'HYUNDAI I22', 'HYUNDAI I26', 'HYUNDAI ALL NEW SANTA FE',
       'HYUNDAI I27'],
      dtype='object')
```

In [15]:

```
#Menampilkan index transmisi
mobil_df.loc[:,"TRANSMISI"].value_counts().index
```

Out[15]:

```
Index(['MANUAL', 'AUTOMATIC'], dtype='object')
```

In [16]:

```
#Filter data dengan mengubah nasiai column dan nows
```

```
#Filter data dengan mengubah posisi column dan rows
#Features yang akan dianalisis adalah tahun, transmisi, merek/model dan harga panduan
#NaN menunjukkan data pada tabel itu kosong atau nilainya 0

mobil_carro_df = mobil_df.pivot_table(index=['TAHUN','TRANSMISI'], columns=['MEREK'], va
lues='HARGA PANDUAN')
mobil_carro_df.head()
```

Out[16]:

	MEREK	CHERY QQ	CHEVROLET	DAIHATSU	DATSUN	DODGE JOURNEY	HYUNDAI	ISUZU	KIA	MITSUBISHI
TAHUN	TRANSMISI									
1999.0	MANUAL	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2000.0	AUTOMATIC	NaN	NaN	NaN	NaN	NaN	NaN	45000000.0	NaN	NaN
	MANUAL	NaN	NaN	9.826258e+07	NaN	NaN	NaN	47000000.0	NaN	NaN
2001.0	AUTOMATIC	NaN	NaN	NaN	NaN	NaN	65000000.0	62500000.0	NaN	NaN
	MANUAL	NaN	NaN	9.826258e+07	NaN	NaN	60000000.0	60000000.0	NaN	NaN

In [0]:

```
#Menyimpan dataset dengan jenis file excel
mobil_carro_df.to_excel("output_mobil.xlsx")
```

Visualisasi Data

Visualisasi data menggunakan seaborn boxplot. Hal ini karena boxplot dapat menunjukkan nilai minimum, maksimum, median, 25 percentile dan 75 percentile. Harapannya dapat diprediksi harga minimum, maksimum dan median untuk setiap merek mobil yang mewakili.

In [18]:

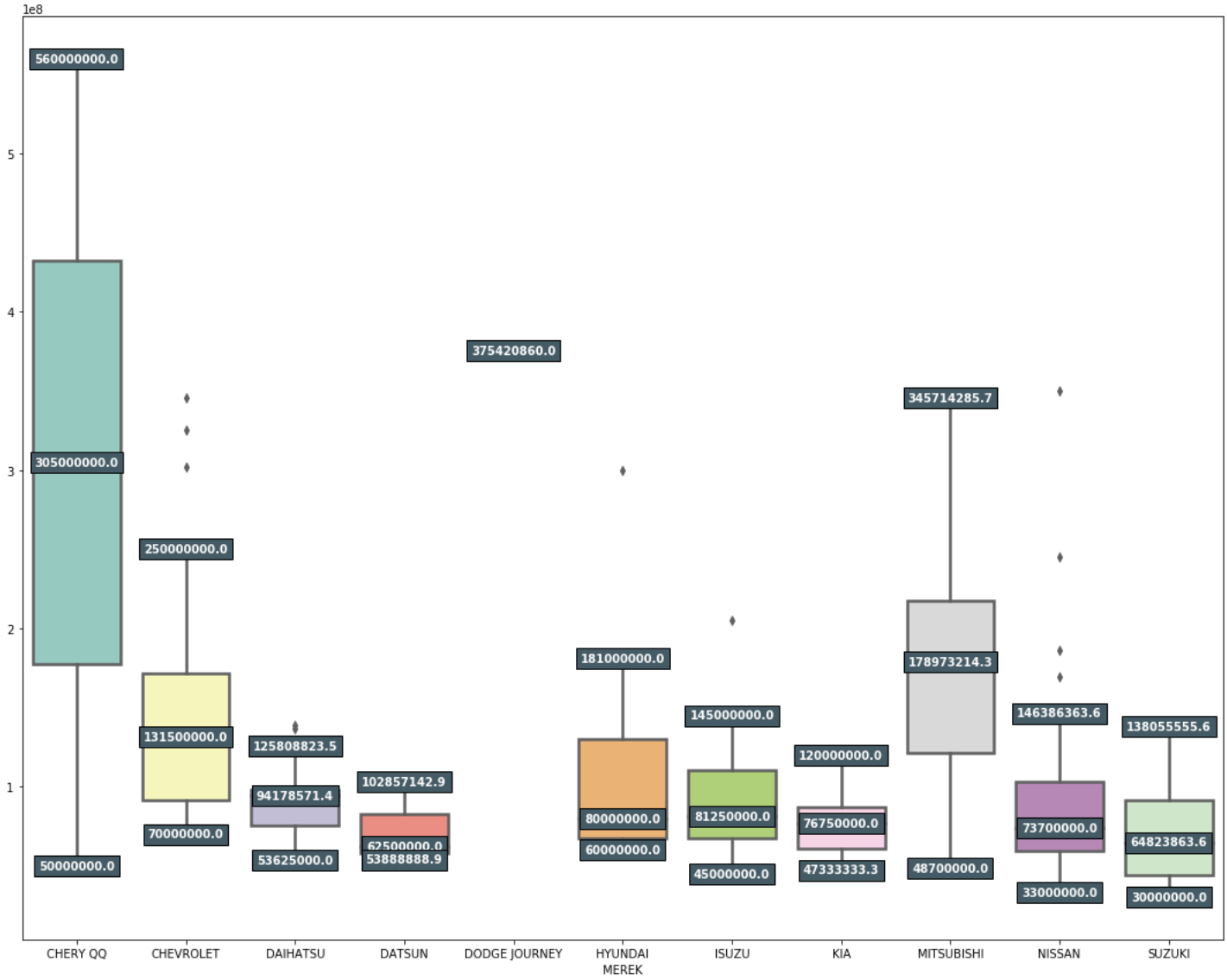
```
import matplotlib
import seaborn as sns
plt.figure(figsize=(15,12))
box_plot = sns.boxplot(data=mobil_carro_df, palette="Set3", linewidth=2.5)

ax = box_plot.axes
lines = ax.get_lines()
categories = ax.get_xticks()

for cat in categories:
    # every 4th line at the interval of 6 is median  line
    x = round(lines[3+cat*6].get_ydata()[0],1)
    y = round(lines[4+cat*6].get_ydata()[0],1)
    z = round(lines[2+cat*6].get_ydata()[0],1)

    ax.text(
        cat,
        x,
        f'{x}',
        ha='center',
        va='center',
        fontweight='bold',
        size=10,
        color='white',
        bbox=dict(facecolor='#445A64'))
    ax.text(
        cat,
        y,
        f'{y}',
        ha='center',
```

```
        va='center',
        fontweight='bold',
        size=10,
        color='white',
        bbox=dict(facecolor='#445A64'))
    ax.text(
        cat,
        z,
        f'{z}',
        ha='center',
        va='center',
        fontweight='bold',
        size=10,
        color='white',
        bbox=dict(facecolor='#445A64'))
#print(x,y,z)
box_plot.figure.tight_layout()
```



Berdasarkan visualisasi data ditunjukkan bahwa terdapat nilai minimum, median, dan maksimum untuk setiap merek mobil. Visualisasi ini ditampilkan secara general. Unt uk dapat menampilkan visualisasi data yang lebih rinci digunakan machine learning sebagai acuan sementara supaya dapat menampilkan data secara konsistem

Pengembangan Tools dan Petunjuk Bagi Manajemen Untuk Penentuan Harga

Salah satu tools yang digunakan adalah dengan menggunakan machine learning. Hal in i karena ketersediaan data yang akan terus bertambah dan bervariasi sehingga dapat memudahkan manajemen untuk menemukan harga yang diharapkan sebagai acuan saja. Seb

enarnya pada kenyataannya harga tersebut dapat disesuaikan dengan keadaan dilapang
an sehingga tidak terlalu berpatok ke machine learning. Bahasa pemograman yang digu
nakan Python alasannya karena saya baru bisa pemograman python. Sebenarnya bisa pa
kai pemograman lain seperti R.

Menentukkan variabel prediksi dan target

In [19]:

```
#Variabel X
#Label yang digunakan tahun, model mobil dan jenis transmisi
#Alasan menggunakan tahun, model dan transmisi karena semakin spesifikasi data yang ditra
ining maka saat dilakukan testing akan memiliki resiko overfitting.
#Namun hal ini dapat disesuaikan dengan kebutuhan
X = mobil_df[['TAHUN', 'MODEL', 'TRANSMISI']].values
X[0:5]
```

Out[19]:

```
array([[2019.0, 'XENIA', 'AUTOMATIC'],
       [2019.0, 'XENIA', 'MANUAL'],
       [2019.0, 'XENIA', 'AUTOMATIC'],
       [2019.0, 'XENIA', 'MANUAL'],
       [2019.0, 'XENIA', 'AUTOMATIC']], dtype=object)
```

In [20]:

```
#Melakukan labeling
#Mengubah string model dan transmisi menjadi float
#Sistem pengkodean dengan mengurutkan dari yang uratan huruf awal terkecil ke besar
from sklearn import preprocessing
le_MODEL = preprocessing.LabelEncoder()
le_MODEL.fit(['XENIA', 'GRAND LIVINA', 'X-TRAIL', 'GRAN MAX', 'APV', 'TERIOS',
       'PANTHER', 'SIRION', 'AYLA', 'SERENA', 'PAJERO SPORT', 'ERTIGA',
       'PICANTO', 'KARIMUN', 'SWIFT', 'CARRY', 'LUXIO', 'MARCH', 'SPLASH',
       'CAPTIVA', 'EVALIA', 'PAJERO', 'KARIMUN WAGON R', 'SIGRA', 'GO+',
       'GRAND VITARA', 'JUKE', 'L300', 'IGNIS', 'X-OVER', 'GO', 'XPANDER',
       'SPIN', 'HYUNDAI ASCENT VERNA', 'KUDA', 'BALENO', 'ARENA',
       'CHEVROLET CAPTIVA', 'CHEVROLET NEW CAPTIVA', 'APV ARENA',
       'CHEVROLET ZAFIRA', 'TARUNA', 'NEO BALENO', 'WAGON R', 'HYUNDAI MATRIX',
       'BALENO NEXT-G', 'KARIMUN WAGON', 'HYUNDAI AVEGA', 'DODGE JOURNEY',
       'CHEVROLET ALL NEW AVEO', 'GO+ PANCA', 'KARIMUN WAGON R GS', 'GO PANCA',
       'APV LUXURY', 'CHERY QQ', 'HYUNDAI GRAND AVEGA', 'HYUNDAI I21',
       'HYUNDAI I26', 'HYUNDAI ALL NEW SANTA FE', 'HYUNDAI I24', 'HYUNDAI I20',
       'HYUNDAI I22', 'HYUNDAI I25', 'HYUNDAI I23', 'HYUNDAI I27'])
X[:,1] = le_MODEL.transform(X[:,1])

#Label 0 (Automatic) dan 1 (Manual)
le_TRANSMISI = preprocessing.LabelEncoder()
le_TRANSMISI.fit(['MANUAL', 'AUTOMATIC'])
X[:,2] = le_TRANSMISI.transform(X[:,2])

X[1409:1420]
```

Out[20]:

```
array([[2014.0, 40, 1],
       [2014.0, 40, 1],
       [2018.0, 41, 1],
       [2018.0, 41, 1],
       [2018.0, 41, 0],
       [2018.0, 41, 0],
       [2018.0, 41, 1],
       [2018.0, 41, 0],
       [2018.0, 41, 1],
       [2016.0, 41, 1],
       [2016.0, 41, 1]], dtype=object)
```

In [21]:

```
#Variabel Y berupa harga panduan
#Variabel Y disebut juga sebagai target
Y = mobil_df['HARGA PANDUAN'].values
Y[0:5]
```

Out[21]:

```
array([1.38e+08, 1.39e+08, 1.40e+08, 1.44e+08, 1.49e+08])
```

Bagi Data Train dan Test

Data train berjumlah 1549 data
Data test berjumlah 664 data
Test size = 0.3
Random state = 1

In [22]:

```
X_trainset, X_testset, Y_trainset, Y_testset = train_test_split(X, Y, test_size=0.3, ran
dom_state=1)
print ('Train set:', X_trainset.shape, Y_trainset.shape)
print ('Test set:', X_testset.shape, Y_testset.shape)
```

```
Train set: (1549, 3) (1549,)
Test set: (664, 3) (664,)
```

Model Fitting Decision Tree Regressor

Pemilihan Model Decision Tree Regressor ini berdasarkan data yang akan dianalisis. Sebenarnya bisa dilakukan dengan menggunakan model mutiple regressi namun inputnya harus berupa angka bukan string. Berdasarkan pengalaman saya, model decision tree dapat digunakan untuk input berupa string tetapi harus dilakukan pelabelan terlebi h dahulu dan biasanya outputnya berupa string. Melihat kendala diatas saya berinisi atif menggabungkan kedua model tersebut menjadi decision tree regressor sehingga ou tputnya dapat berupa angka/value

In [23]:

```
#Training data dilakukan menggunakan dicision tree regressor
#Model fit berupa X_trainset dan Y_trainset
#Kriteria yang digunakan "mse"
regressor = DecisionTreeRegressor()
regressor.fit(X_trainset, Y_trainset)
```

Out[23]:

```
DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
                      max_leaf_nodes=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      presort=False, random_state=None, splitter='best')
```

Test Dataset

In [24]:

```
#Untuk test data
#Kolom 1 : tahun mobil
#Kolom 2 : mpdel mobil
#Kolom 3 : transmisi mobil
X_testset[0:5]
```

Out[24]:

```
array([[2017.0, 40, 0],
       [2014.0, 0, 0],
       [2006.0, 52, 0],
       [2011.0, 59, 0],
       [2008.0, 63, 1]], dtype=object)
```

In [25]:

```
#Hasil Y prediksi
y_pred = regressor.predict(X_testset)
y_pred[0:5]
```

Out[25]:

```
array([70000000., 92000000., 56200000., 99187500., 57750000.])
```

In [26]:

```
#Membandingkan data Aktual dan Prediksi
df=pd.DataFrame({'Actual':Y_testset, 'Prediksi':y_pred})
df.head()
```

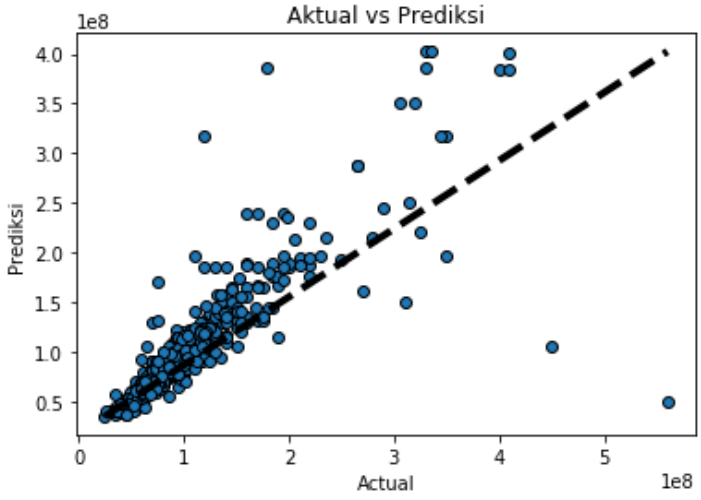
Out[26]:

	Actual	Prediksi
0	73000000.0	70000000.0
1	92000000.0	92000000.0
2	56000000.0	56200000.0
3	96000000.0	99187500.0
4	55000000.0	57750000.0

In [27]:

```
#Grafik Aktual dan Prediksi
from sklearn.model_selection import cross_val_predict

fig, ax = plt.subplots()
ax.scatter(Y_testset, y_pred, edgecolors=(0, 0, 0))
ax.plot([Y_testset.min(), Y_testset.max()], [y_pred.min(), y_pred.max()], 'k--', lw=4)
ax.set_xlabel('Actual')
ax.set_ylabel('Prediksi')
ax.set_title("Aktual vs Prediksi")
plt.show()
```



Akurasi Data

In [28]:

```
from sklearn import metrics
```

```
print('Mean Absolute Error:', metrics.mean_absolute_error(Y_testset, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(Y_testset, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(Y_testset, y_pred)))

from sklearn.metrics import r2_score
print("R2-score: %.2f" % (r2_score(y_pred , Y_testset)))
```

```
Mean Absolute Error: 12444823.652597958
Mean Squared Error: 1056756020971690.2
Root Mean Squared Error: 32507784.00586066
R2-score: 0.64
```

Akurasi yang diperoleh dari model yang digunakan 64%. Sebenarnya masih rendah. Hal ini dapat ditingkatkan dengan menambah data sehingga data yang digunakan harus banyak. Sebenarnya ini salah satu kelemahan machine learning sehingga data yang digunakan harus banyak sehingga tingkat keakurasian data bertambah. Variasi data juga menentukan akurasi data.

Rekomendasi Untuk Penyempurnaan Sistem Prediksi

Rekomendasi untuk penyempurnaan sistem prediksi harga dan SOP serta tools yang diperlukan untuk memelihara akurasi prediksi terdiri dari:

1. Menambah data pada model mobil yang telah ada
2. Pengembangan jauh kedepannya dapat dilakukan metode deep learning sehingga dari foto mobil dapat dianalisis feature model mobil, merek mobil, tahun mobil dan dapat dikorelasikan dengan harga namuan ketersediaan datanya harus berjuta-juta sehingga akurasinya dapat meningkat.