```
import numpy as np
import pandas as pd
```

```
df=pd.read_csv('used_cars_dataset.csv')
df.head()
```

	name	year	km_driven	fuel	seller_type	transmission	owner	selling_price
0	Maruti 800 AC	2007	70000	Petrol	Individual	Manual	First Owner	60000
1	Maruti Wagon R LXI Minor	2007	50000	Petrol	Individual	Manual	First Owner	135000
2	Hyundai Verna 1.6 SX	2012	100000	Diesel	Individual	Manual	First Owner	600000
3	Datsun RediGO T Option	2017	46000	Petrol	Individual	Manual	First Owner	250000
4	Honda Amaze VX i-DTEC	2014	141000	Diesel	Individual	Manual	Second Owner	450000

Next steps: Generate code with df | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4340 entries, 0 to 4339
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   name           4340 non-null   object
 1   year           4340 non-null   int64
 2   km_driven      4340 non-null   int64
 3   fuel           4340 non-null   object
 4   seller_type    4340 non-null   object
 5   transmission   4340 non-null   object
 6   owner          4340 non-null   object
 7   selling_price  4340 non-null   int64
dtypes: int64(3), object(5)
memory usage: 271.4+ KB
```

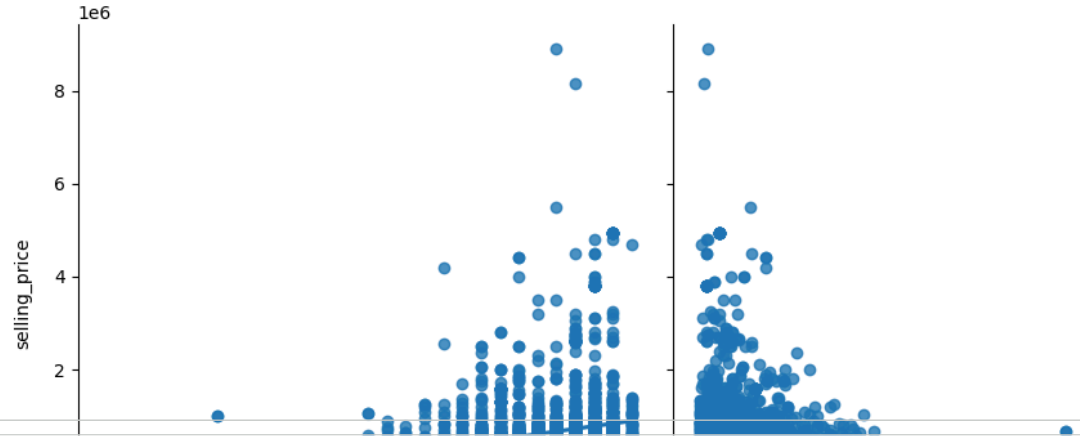
```
df.isnull().sum()
```

	0
name	0
year	0
km_driven	0
fuel	0
seller_type	0
transmission	0
owner	0
selling_price	0

dtype: int64

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.pairplot(data=df,
             x_vars=['year','km_driven'],
             y_vars=['selling_price'],
             height=5,
             aspect=1,
             kind='reg')
plt.show()
```




```
from sklearn.preprocessing import StandardScaler,OneHotEncoder
from sklearn.compose import ColumnTransformer

numerical_col=df.select_dtypes(include=np.number)
categorical_col=df.select_dtypes(exclude=np.number)

preprocessor=ColumnTransformer(transformers=[
    ('num',StandardScaler(),numerical_col.columns),
    ('cat',OneHotEncoder(handle_unknown='ignore'),categorical_col.columns)
])

x=preprocessor.fit_transform(df)
y=df['selling_price']
```

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
```

▾ LinearRegression

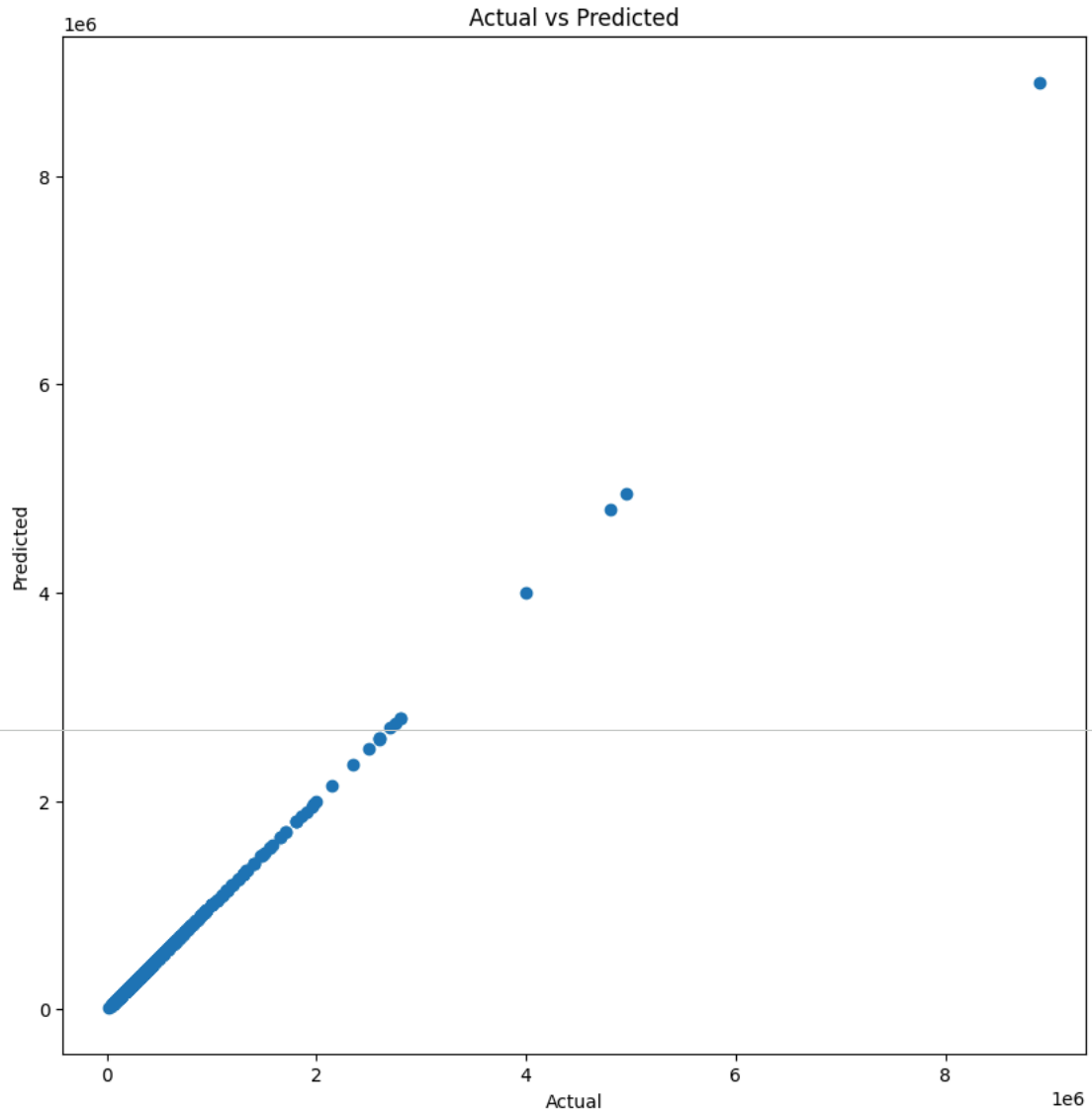
```
LinearRegression()
```

```
y_pred=model.predict(x_test)
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error

print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
print(mean_squared_error(y_test,y_pred))
```

```
0.9999999999577959
2.352358972232948
12.879448112095707
```

```
plt.figure(figsize=(10,10))
plt.scatter(y_test,y_pred)
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs Predicted')
plt.show()
```


Actual vs Predicted
1e6
Predicted
Actual
1e6
0
2
4
6
8
0
2
4
6
8