AI and ML task 3

Objectives

Implement and understand simple & multiple linear regression. 1.Import and preprocess the dataset. 2.Split data into train-test sets. 3.Fit a Linear Regression model using sklearn.linear_model. 4.Evaluate model using MAE, MSE, R². 5.Plot regression line and interpret coefficients

Importing libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
from sklearn.linear_model import LinearRegression
```

In [ ]:
```

```

In [3]:
```
data = pd.read_csv('Housing.csv')
```

In [11]:
```
data
```

Out[11]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished
...	...	...	...	...	...	...	...	...	...	...	...	...	...
540	1820000	3000	2	1	1	yes	no	yes	no	no	2	no	unfurnished
541	1767150	2400	3	1	1	no	no	no	no	no	0	no	semi-furnished
542	1750000	3620	2	1	1	yes	no	no	no	no	0	no	unfurnished
543	1750000	2910	3	1	1	no	no	no	no	no	0	no	furnished
544	1750000	3850	3	1	2	yes	no	no	no	no	0	no	unfurnished

545 rows × 13 columns

In [13]:
```
data.shape
```

Out[13]:
```
(545, 13)
```

In [7]:
```
data.describe()
```

Out[7]:

	price	area	bedrooms	bathrooms	stories	parking
count	5.450000e+02	545.000000	545.000000	545.000000	545.000000	545.000000
mean	4.766729e+06	5150.541284	2.965138	1.286239	1.805505	0.693578
std	1.870440e+06	2170.141023	0.738064	0.502470	0.867492	0.861586
min	1.750000e+06	1650.000000	1.000000	1.000000	1.000000	0.000000
25%	3.430000e+06	3600.000000	2.000000	1.000000	1.000000	0.000000
50%	4.340000e+06	4600.000000	3.000000	1.000000	2.000000	0.000000
75%	5.740000e+06	6360.000000	3.000000	2.000000	2.000000	1.000000
max	1.330000e+07	16200.000000	6.000000	4.000000	4.000000	3.000000

In [9]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

Check for missing values

In [63]:
```
missing_values = data.isnull().sum()
```

In [67]:
```
missing_values
```

Out[67]:
```
price                            0
area                             0
bedrooms                         0
bathrooms                        0
stories                          0
mainroad                         0
guestroom                        0
basement                         0
hotwaterheating                  0
airconditioning                  0
parking                          0
prefarea                         0
furnishingstatus_semi-furnished  0
furnishingstatus_unfurnished     0
dtype: int64
```

Handling categorcal data

In [23]:
```
categorical_cols = ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea', 'furnishingstatus']
```

In [27]:
```
binary_mapping = {'yes': 1, 'no': 0}
for col in categorical_cols[:-1]:  # all except 'furnishingstatus'
    data[col] = data[col].map(binary_mapping)
```

In [33]:
```
data = pd.get_dummies(data, columns=['furnishingstatus'], drop_first=True)
data.head(), missing_values
```

Out[33]:
```
(      price  area  bedrooms  bathrooms  stories  mainroad  guestroom  \
 0  13300000  7420         4          2        3         1          0   
 1  12250000  8960         4          4        4         1          0   
 2  12250000  9960         3          2        2         1          0   
 3  12215000  7500         4          2        2         1          0   
 4  11410000  7420         4          1        2         1          1   

    basement  hotwaterheating  airconditioning  parking  prefarea  \
 0         0                0                1        2         1   
 1         0                0                1        3         0   
 2         1                0                0        2         1   
 3         1                0                1        3         1   
 4         1                0                1        2         0   

   furnishingstatus_semi-furnished  furnishingstatus_unfurnished  
 0                            False                         False  
 1                            False                         False  
 2                             True                         False  
 3                            False                         False  
 4                            False                         False  ,
 price                            0
 area                             0
 bedrooms                         0
 bathrooms                        0
 stories                          0
 mainroad                         0
 guestroom                        0
 basement                         0
 hotwaterheating                  0
 airconditioning                  0
 parking                          0
 prefarea                         0
 furnishingstatus                 0
 dtype: int64)
```

Fit a Linear Regression model using sklearn.linear_model.

In [36]:
```
from sklearn.model_selection import train_test_split
```

In [38]:
```
X = data.drop('price', axis=1)
y = data['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
train_score = lr_model.score(X_train, y_train)
test_score = lr_model.score(X_test, y_test)

train_score, test_score
```

Out[38]:
```
(0.6859438988560158, 0.6529242642153184)
```

In [42]:
```
print("Training R^2 Score:", lr_model.score(X_train, y_train))
print("Testing R^2 Score:", lr_model.score(X_test, y_test))
```

```
Training R^2 Score: 0.6859438988560158
Testing R^2 Score: 0.6529242642153184
```

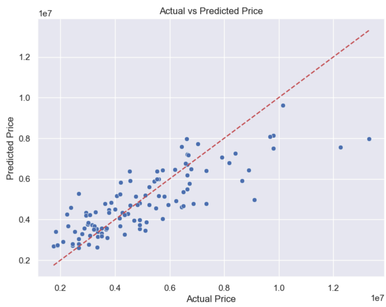
Evaluate model using MAE, MSE, R².

In [48]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
y_pred = lr_model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R² Score: {r2:.4f}")
```

```
Mean Absolute Error (MAE): 970043.40
Mean Squared Error (MSE): 1754318687330.66
R² Score: 0.6529
```

Plot regression line and interpret coefficients.

In [50]:
```
y_pred = lr_model.predict(X_test)
plt.figure(figsize=(8,6))
sns.scatterplot(x=y_test, y=y_pred)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')  # perfect prediction line
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual vs Predicted Price')
plt.grid(True)
plt.show()
```


Actual vs Predicted Price
1e7
1.2
1.0
0.8
0.6
0.4
0.2
Predicted Price
0.2
0.4
0.6
0.8
1.0
1.2
1e7
Actual Price

In [ ]: