

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error

import joblib
```

```
from google.colab import drive
drive.mount('drive')

Drive already mounted at drive; to attempt to forcibly remount, call drive.mount("drive", force_remount=True).
```

```
from google.colab import files
files.upload()
```

[Show hidden output](#)

```
import os
os.listdir()

['.config', 'drive', 'sample_data']

os.rename("india_housing_prices.csv.csv", "india_housing_prices.csv")
```

```
df = pd.read_csv("india_housing_prices.csv")

print(df.head())
print(df.shape)

      ID      State     City    Locality  Property_Type  BHK \
0    1  Tamil Nadu  Chennai  Locality_84      Apartment    1
1    2  Maharashtra  Pune  Locality_490  Independent House    3
2    3      Punjab  Ludhiana  Locality_167      Apartment    2
3    4  Rajasthan  Jodhpur  Locality_393  Independent House    2
4    5  Rajasthan  Jaipur  Locality_466        Villa    4

   Size_in_SqFt  Price_in_Lakhs  Price_per_SqFt  Year_Built  ... \
0       4740        489.76         0.10      1990  ...
1       2364        195.52         0.08      2008  ...
2       3642        183.79         0.05      1997  ...
3       2741        300.29         0.11      1991  ...
4       4823        182.90         0.04      2002  ...

  Age_of_Property  Nearby_Schools  Nearby_Hospitals \
0            35                  10                   3
1            17                   8                   1
2            28                   9                   8
3            34                   5                   7
4            23                   4                   9

  Public_Transport_Accessibility  Parking_Space  Security \
0                      High             No           No
1                      Low             No           Yes
2                      Low            Yes           No
3                      High            Yes           Yes
4                      Low             No           Yes

  Amenities Facing Owner_Type \
0  Playground, Gym, Garden, Pool, Clubhouse    West      Owner
1  Playground, Clubhouse, Pool, Gym, Garden   North     Builder
2  Clubhouse, Pool, Playground, Gym          South     Broker
3  Playground, Clubhouse, Gym, Pool, Garden   North     Builder
4  Playground, Garden, Gym, Pool, Clubhouse  East     Builder

  Availability_Status
0      Ready_to_Move
1  Under_Construction
2      Ready_to_Move
3      Ready_to_Move
4      Ready_to_Move

[5 rows x 23 columns]
(250000, 23)
```

```
df = df.drop_duplicates()
df = df.fillna(method="ffill")
```

```
/tmp/ipython-input-2895020615.py:2: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a future versi
df = df.fillna(method="ffill")
```

```
numeric_df = df.select_dtypes(include=['int64', 'float64'])

corr_matrix = numeric_df.corr()

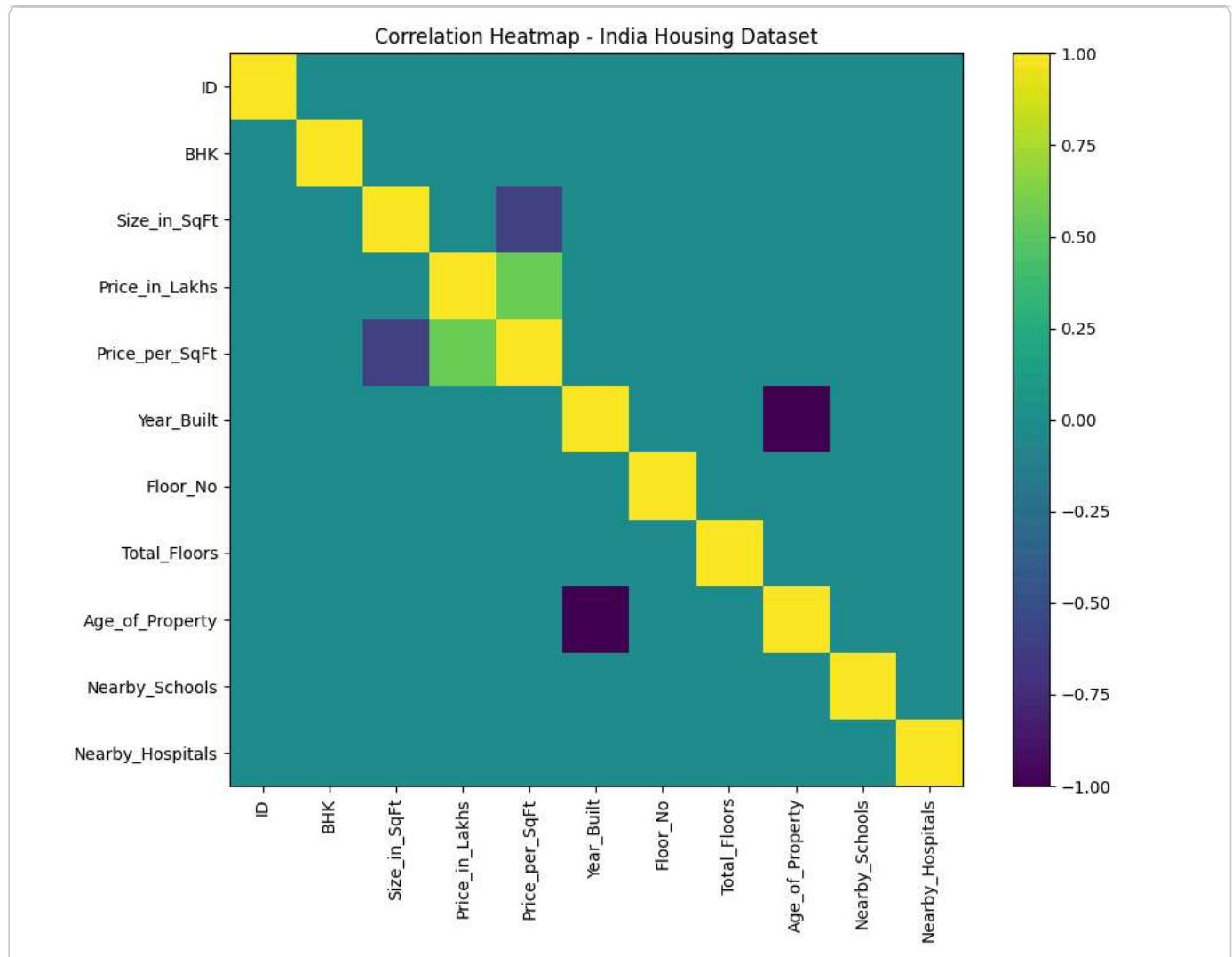
plt.figure(figsize=(10,8))

plt.imshow(corr_matrix)
plt.colorbar()

plt.xticks(range(len(corr_matrix.columns)), corr_matrix.columns, rotation=90)
plt.yticks(range(len(corr_matrix.columns)), corr_matrix.columns)

plt.title("Correlation Heatmap - India Housing Dataset")

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



```
from google.colab import drive
drive.mount('drive')
```

```
Drive already mounted at drive; to attempt to forcibly remount, call drive.mount("drive", force_remount=True).
```

```
from google.colab import drive
drive.mount('drive')
```

```
Drive already mounted at drive; to attempt to forcibly remount, call drive.mount("drive", force_remount=True).
```

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

ID	0
State	0
City	0
Locality	0
Property_Type	0
BHK	0
Size_in_SqFt	0
Price_in_Lakhs	0
Price_per_SqFt	0
Year_Built	0
Furnished_Status	0
Floor_No	0
Total_Floors	0
Age_of_Property	0
Nearby_Schools	0
Nearby_Hospitals	0
Public_Transport_Accessibility	0
Parking_Space	0
Security	0
Amenities	0
Facing	0

```
Owner_Type          0
Availability_Status 0
dtype: int64
```

```
df = df.dropna()
```

```
print(df.info())
print(df.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250000 entries, 0 to 249999
Data columns (total 23 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   ID               250000 non-null   int64  
 1   State            250000 non-null   object  
 2   City              250000 non-null   object  
 3   Locality          250000 non-null   object  
 4   Property_Type    250000 non-null   object  
 5   BHK              250000 non-null   int64  
 6   Size_in_SqFt     250000 non-null   int64  
 7   Price_in_Lakhs   250000 non-null   float64
 8   Price_per_SqFt   250000 non-null   float64
 9   Year_Built       250000 non-null   int64  
 10  Furnished_Status 250000 non-null   object  
 11  Floor_No         250000 non-null   int64  
 12  Total_Floors    250000 non-null   int64  
 13  Age_of_Property 250000 non-null   int64  
 14  Nearby_Schools   250000 non-null   int64  
 15  Nearby_Hospitals 250000 non-null   int64  
 16  Public_Transport_Accessibility 250000 non-null   object  
 17  Parking_Space    250000 non-null   object  
 18  Security          250000 non-null   object  
 19  Amenities         250000 non-null   object  
 20  Facing             250000 non-null   object  
 21  Owner_Type        250000 non-null   object  
 22  Availability_Status 250000 non-null   object  
dtypes: float64(2), int64(9), object(12)
memory usage: 43.9+ MB
None
```

	ID	BHK	Size_in_SqFt	Price_in_Lakhs
count	250000.000000	250000.000000	250000.000000	250000.000000
mean	125000.500000	2.999396	2749.813216	254.586854
std	72168.927986	1.415521	1300.606954	141.349921
min	1.000000	1.000000	500.000000	10.000000
25%	62500.750000	2.000000	1623.000000	132.550000
50%	125000.500000	3.000000	2747.000000	253.870000
75%	187500.250000	4.000000	3874.000000	376.880000
max	250000.000000	5.000000	5000.000000	500.000000

	Price_per_SqFt	Year_Built	Floor_No	Total_Floors
count	250000.000000	250000.000000	250000.000000	250000.000000
mean	0.130597	2006.520012	14.966800	15.503004
std	0.130747	9.808575	8.948047	8.671618
min	0.000000	1990.000000	0.000000	1.000000
25%	0.050000	1998.000000	7.000000	8.000000
50%	0.090000	2007.000000	15.000000	15.000000
75%	0.160000	2015.000000	23.000000	23.000000
max	0.990000	2023.000000	30.000000	30.000000

```
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( ① ② )  
LinearRegression()
```

```
y_pred = model.predict(X_test)
```

```
from sklearn.metrics import r2_score  
  
accuracy = r2_score(y_test, y_pred)  
  
print("Model Accuracy:", accuracy)
```

```
Model Accuracy: 0.4901544795453481
```

```
plt.figure(figsize=(6,6))  
plt.scatter(y_test, y_pred)  
plt.xlabel("Actual Price")  
plt.ylabel("Predicted Price")  
plt.title("Actual vs Predicted House Prices")  
plt.show()
```



```
import joblib
```

```
joblib.dump(model, "house_price_model.pkl")  
['house_price_model.pkl']  
  
loaded_model = joblib.load("house_price_model.pkl")
```

```
# Take a sample row (keeps column structure)  
sample_input = X.iloc[[0]]  
  
prediction = loaded_model.predict(sample_input)  
  
print("Predicted House Price:", prediction[0])
```

```
Predicted House Price: 343.63475143758785
```

```
coeff = pd.DataFrame(model.coef_, X.columns)  
print(coeff)
```

	0
ID	0.000003
State	0.020079
City	0.008974
Locality	0.001453
Property_Type	-0.476551
BHK	-0.115375
Size_in_SqFt	0.059257
Price_per_SqFt	962.326009
Year_Built	0.005277
Furnished_Status	-0.092937
Floor_No	0.021078
Total_Floors	-0.003408
Age_of_Property	-0.005277
Nearby_Schools	-0.037425
Nearby_Hospitals	-0.068850
Public_Transport_Accessibility	-0.197173
Parking_Space	0.173404
Security	0.719424
Amenities	0.003331
Facing	-0.049863
Owner_Type	-0.294903
Availability_Status	-0.197265