importing modules and dataset

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

```
df=pd.read_csv("C:\\Users\\hp\\Downloads\\prodigy infotech\\housing_price_dataset.csv")
df.head()
```

	SquareFeet	Bedrooms	Bathrooms	Neighborhood	YearBuilt	Price
0	2126	4	1	Rural	1969	215355.283618
1	2459	3	2	Rural	1980	195014.221626
2	1860	2	1	Suburb	1970	306891.012076
3	2294	2	1	Urban	1996	206786.787153
4	2130	5	2	Suburb	2001	272436.239065

data preprocessing and eda

```
df.info() #to check basic info of the data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   SquareFeet    50000 non-null  int64
 1   Bedrooms      50000 non-null  int64
 2   Bathrooms     50000 non-null  int64
 3   Neighborhood  50000 non-null  object
 4   YearBuilt     50000 non-null  int64
 5   Price         50000 non-null  int32
dtypes: int32(1), int64(4), object(1)
memory usage: 2.1+ MB
```

converting float to int of Price column

```
df['Price']=df['Price'].astype(int)
df.info() #to check info of the data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   SquareFeet    50000 non-null  int64
 1   Bedrooms      50000 non-null  int64
 2   Bathrooms     50000 non-null  int64
 3   Neighborhood  50000 non-null  object
 4   YearBuilt     50000 non-null  int64
 5   Price         50000 non-null  int32
dtypes: int32(1), int64(4), object(1)
memory usage: 2.1+ MB
```

convert categorical column to int

```
df['Neighborhood'].unique() #to check unique values
```

```
array(['Rural', 'Suburb', 'Urban'], dtype=object)
```

```
df1=pd.get_dummies(df['Neighborhood'])
df1.head() #to print dummies values
```

	Rural	Suburb	Urban
0	1	0	0
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

```
df1.drop('Rural',axis=1,inplace=True) #drop useless columns
```

```
df1.head()
```

	Suburb	Urban
0	0	0
1	0	0
2	1	0
3	0	1
4	1	0

```
final=pd.concat([df,df1],axis=1)
final.head() #to merge both columns
```

	SquareFeet	Bedrooms	Bathrooms	Neighborhood	YearBuilt	Price	Suburb	Urban
0	2126	4	1	Rural	1969	215355	0	0
1	2459	3	2	Rural	1980	195014	0	0
2	1860	2	1	Suburb	1970	306891	1	0
3	2294	2	1	Urban	1996	206786	0	1
4	2130	5	2	Suburb	2001	272436	1	0

```
final.drop('Neighborhood',axis=1,inplace=True) #drop useless columns
```

```
final.head()
```

	SquareFeet	Bedrooms	Bathrooms	YearBuilt	Price	Suburb	Urban
0	2126	4	1	1969	215355	0	0
1	2459	3	2	1980	195014	0	0
2	1860	2	1	1970	306891	1	0
3	2294	2	1	1996	206786	0	1
4	2130	5	2	2001	272436	1	0

```
final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   SquareFeet  50000 non-null  int64
 1   Bedrooms    50000 non-null  int64
 2   Bathrooms   50000 non-null  int64
 3   YearBuilt   50000 non-null  int64
 4   Price       50000 non-null  int32
 5   Suburb      50000 non-null  uint8
 6   Urban       50000 non-null  uint8
dtypes: int32(1), int64(4), uint8(2)
memory usage: 1.8 MB
```

```
final['Suburb']=final['Suburb'].astype(int)
final['Urban']=final['Urban'].astype(int)
```

```
final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   SquareFeet  50000 non-null  int64
 1   Bedrooms    50000 non-null  int64
 2   Bathrooms   50000 non-null  int64
 3   YearBuilt   50000 non-null  int64
 4   Price       50000 non-null  int32
 5   Suburb      50000 non-null  int32
 6   Urban       50000 non-null  int32
dtypes: int32(3), int64(4)
memory usage: 2.1 MB
```

```
from sklearn.linear_model import LinearRegression
```

```
lr=LinearRegression()
```

```
from sklearn.model_selection import train_test_split
```

```
X=final.drop('Price',axis=1)
y=final['Price']
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
```

```
X_train.head()
```

	SquareFeet	Bedrooms	Bathrooms	YearBuilt	Suburb	Urban
39087	2498	2	3	2021	0	0
30893	2380	5	3	1977	0	1
45278	2274	5	2	1957	0	1
16398	2215	5	1	1977	1	0
13653	2078	2	3	1962	1	0

```
final[39086:39088:1]
```

	SquareFeet	Bedrooms	Bathrooms	YearBuilt	Price	Suburb	Urban
39086	1856	2	3	1970	191313	0	0
39087	2498	2	3	2021	288178	0	0

```
len(X_train)
```

```
40000
```

```
len(X_test)
```

```
10000
```

```
from sklearn.linear_model import LinearRegression
```

```
lr=LinearRegression()
```

```
lr.fit(X_train,y_train)
```

```
LinearRegression()
```

```
print("test score :",lr.score(X_test,y_test))
print("train score :",lr.score(X_train,y_train))
```

```
test score : 0.5755628291469783
train score : 0.5688922008119348
```

```
# this is the model that is used to predict the prices of
# houses based on their square footage and the number of bedrooms and bathrooms.
```

```
lr.predict([[2498,2,3,2021,0,0]]) #bedroom = 2 , bathroom =3
```

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was
  warnings.warn(
array([268316.54483847])
```

```
lr.predict([[2380,5,3,1977,0,1]]) #bedroom =5 , bathroom =2
```

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was
  warnings.warn(
array([273952.08453754])
```

Start coding or generate with AI.