In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data=pd.read_csv(r"C:\Users\user\Downloads\10_USA_Housing.csv")
data
```

Out[2]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Ad
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Fer 674\nLaurabu
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Suite 079 Kathleer
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Eli Stravenue\nDanie WI 0
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nF
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond AE
...	...	...	...	...	...	...	
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams AP 3015
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 925 8489\nAPO AA
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy C Suite 076\nJoshu V
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nF
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George I Apt. 509\nEas

5000 rows × 7 columns

In [3]:
```
df=data.head(100)
df
```

Out[3]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Addı
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry 674\nLaurabury 37(
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Vi Suite 079\nL Kathleen, (
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizal Stravenue\nDanielt WI 064
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPC 44
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nF AE 09
...	...	...	...	...	...	...	
95	73698.696357	6.372731	6.578352	4.33	51815.096419	1.840236e+06	69600 Wi Rue\nElizabethl PW 17767-2
96	66263.910501	6.374930	6.736974	4.41	49829.951500	1.441422e+06	001 S Plaza\nJessicastad 25
97	75394.759582	6.933352	7.834100	4.48	22268.075306	1.498641e+06	238 Anth Drive\nAcostaha AS 62753-6
98	65984.750645	6.512270	6.316480	2.38	21867.476940	9.547466e+05	01818 Wi Spurs\nV Andreaton, SD 69 7
99	66477.262792	6.982152	6.038488	2.37	32458.986699	1.124636e+06	95779 Wa Square\nBakersl PR 49

100 rows × 7 columns

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Avg. Area Income              100 non-null    float64
 1   Avg. Area House Age           100 non-null    float64
 2   Avg. Area Number of Rooms     100 non-null    float64
 3   Avg. Area Number of Bedrooms  100 non-null    float64
 4   Area Population               100 non-null    float64
 5   Price                         100 non-null    float64
 6   Address                       100 non-null    object 
dtypes: float64(6), object(1)
memory usage: 5.6+ KB
```

In [5]:
```
df.describe()
```

Out[5]:

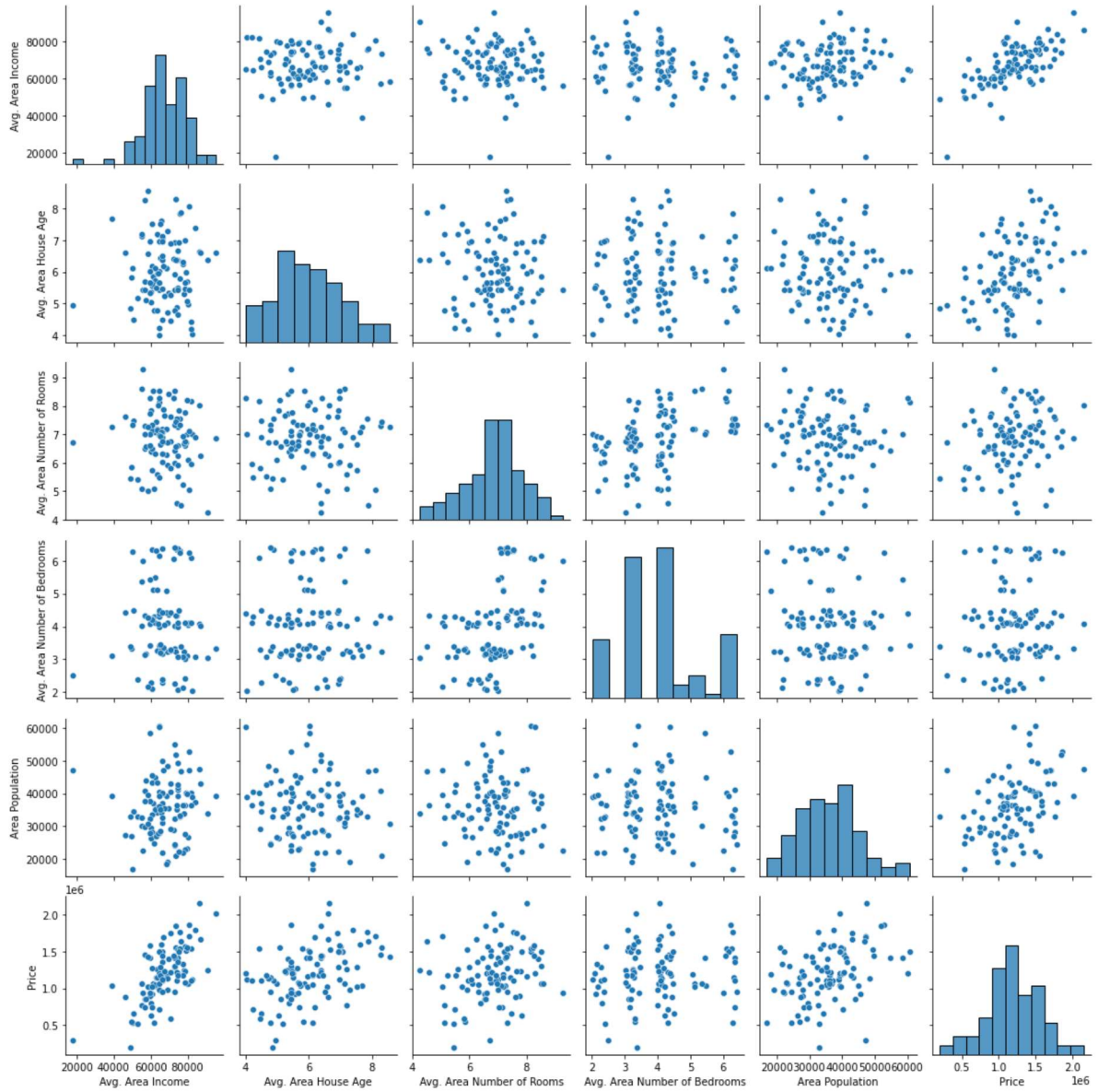
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
count	100.000000	100.000000	100.000000	100.000000	100.000000	1.000000e+02
mean	67419.354519	6.017796	6.860572	4.004600	35918.684716	1.197750e+06
std	11231.604384	1.032219	0.992326	1.197617	9243.180363	3.622928e+05
min	17796.631190	4.010907	4.242191	2.030000	16810.783311	2.018981e+05
25%	60800.655369	5.343417	6.302292	3.195000	29172.345270	1.009614e+06
50%	66408.895319	5.948502	6.954229	4.040000	35620.473543	1.181989e+06
75%	74544.868943	6.640431	7.447286	4.422500	40517.769280	1.444560e+06
max	95450.293086	8.562611	9.289854	6.410000	60828.249085	2.146925e+06

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Room
s',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Addres
s'],
      dtype='object')
```

In [7]:
```
sns.pairplot(df)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x134bebf8d90>
```



In [8]:
```
da=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price']]
da
```

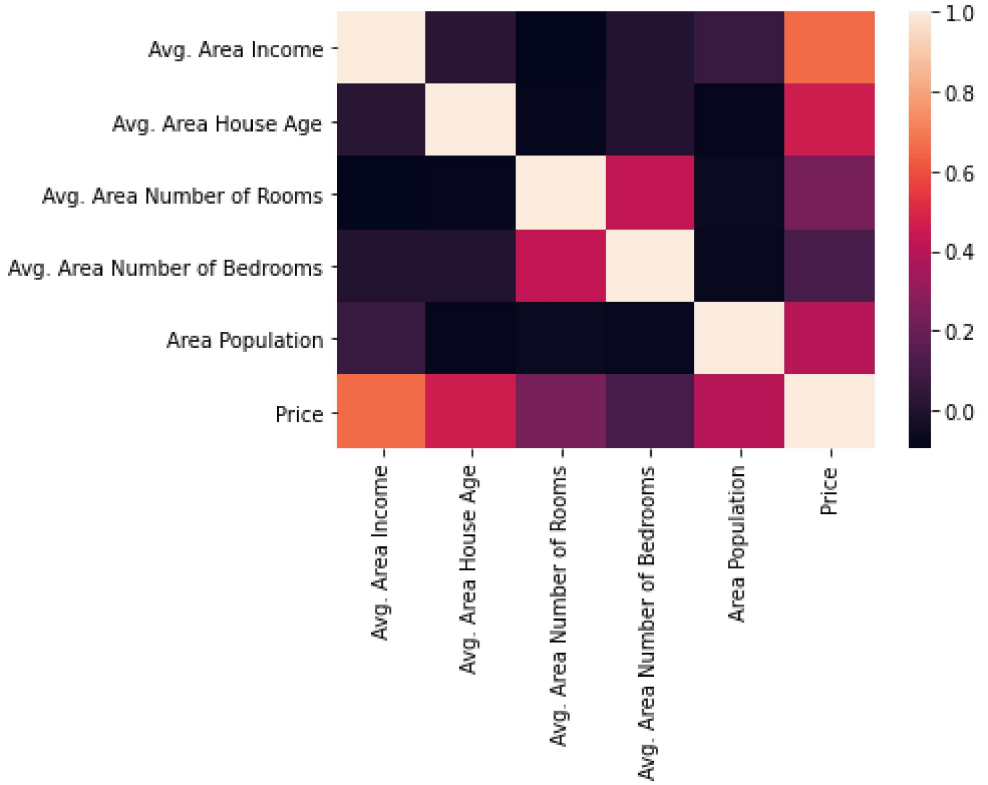
Out[8]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05
...	...	...	...	...	...	...
95	73698.696357	6.372731	6.578352	4.33	51815.096419	1.840236e+06
96	66263.910501	6.374930	6.736974	4.41	49829.951500	1.441422e+06
97	75394.759582	6.933352	7.834100	4.48	22268.075306	1.498641e+06
98	65984.750645	6.512270	6.316480	2.38	21867.476940	9.547466e+05
99	66477.262792	6.982152	6.038488	2.37	32458.986699	1.124636e+06

100 rows × 6 columns

In [9]:
```
sns.heatmap(da.corr())
```

Out[9]: <AxesSubplot:>



In [10]:
```
x=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population']]
y=df['Price']
```

In [11]:
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [12]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[12]:
```
LinearRegression()
```

In [13]:
```
print(lr.intercept_)
```

```
-2705255.3129567625
```

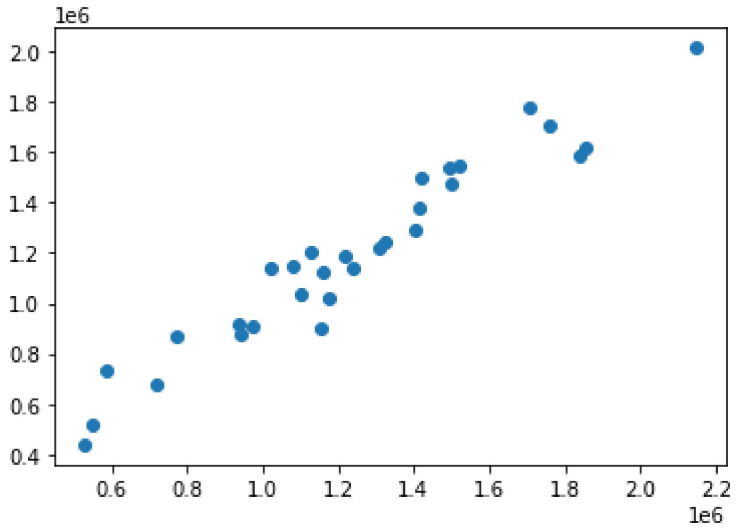
In [14]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[14]:

	Co-efficient
Avg. Area Income	20.573829
Avg. Area House Age	185677.141099
Avg. Area Number of Rooms	128325.370372
Avg. Area Number of Bedrooms	-12050.703754
Area Population	15.421778

In [15]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]: <matplotlib.collections.PathCollection at 0x134c227a940>



In [16]:
```
print(lr.score(x_test,y_test))
```

```
0.9206102508107372
```

In [17]:
```
print(lr.score(x_train,y_train))
```

```
0.9020038095115043
```

In [18]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [19]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[19]: Ridge(alpha=10)

In [20]:
```
rr.score(x_test,y_test)
```

Out[20]: 0.9173916645121715

In [21]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[21]: Lasso(alpha=10)

In [22]:
```
la.score(x_test,y_test)
```

Out[22]: 0.9206151912339886

In [ ]: