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

In [2]:
```
df=pd.read_csv("10_USA_Housing.csv")
df
```

Out[2]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Add
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferr 674\nLaurabur 3
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson \ Suite 079\r Kathleen,
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Eliz Stravenue\nDaniel WI 06
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFP 4
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 30153
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258 8489\nAPO AA 4
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Ga Suite 076\nJoshua VA
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFP 7
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George R Apt. 509\nEast N

5000 rows × 7 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Avg. Area Income              5000 non-null   float64
 1   Avg. Area House Age           5000 non-null   float64
 2   Avg. Area Number of Rooms     5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population               5000 non-null   float64
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object 
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
```

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

Out[4]:

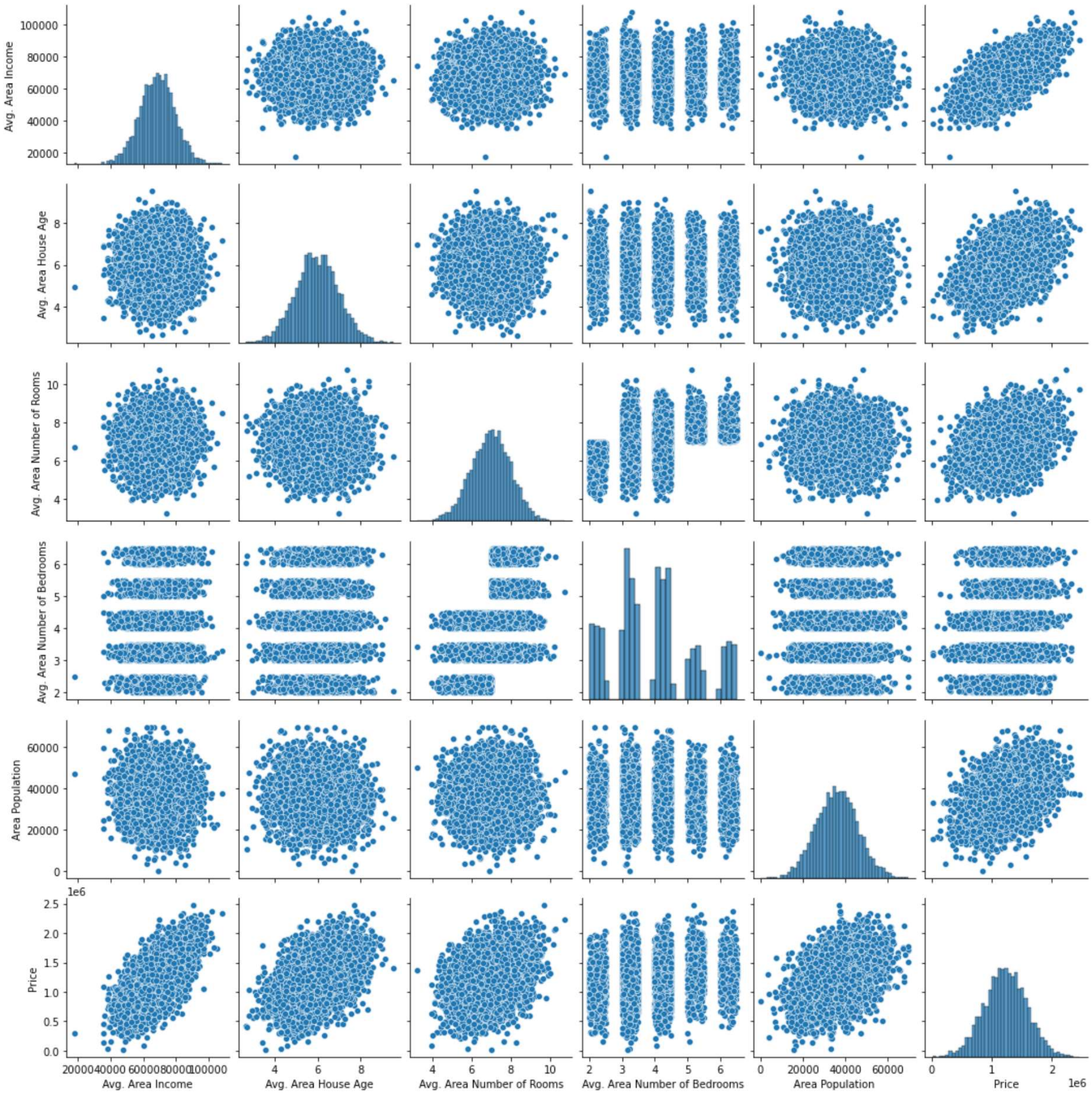
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.000000e+03
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.232073e+06
std	10657.991214	0.991456	1.005833	1.234137	9925.650114	3.531176e+05
min	17796.631190	2.644304	3.236194	2.000000	172.610686	1.593866e+04
25%	61480.562388	5.322283	6.299250	3.140000	29403.928702	9.975771e+05
50%	68804.286404	5.970429	7.002902	4.050000	36199.406689	1.232669e+06
75%	75783.338666	6.650808	7.665871	4.490000	42861.290769	1.471210e+06
max	107701.748378	9.519088	10.759588	6.500000	69621.713378	2.469066e+06

In [5]:

```
sns.pairplot(df)
```

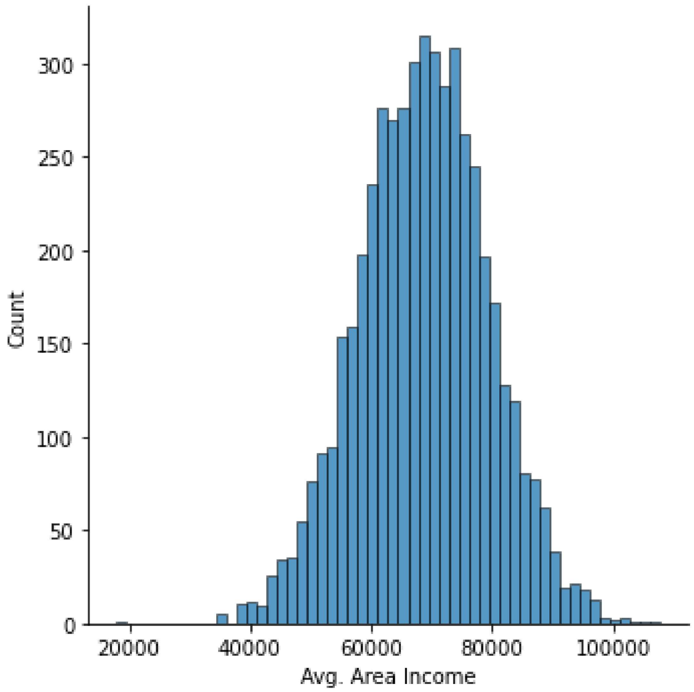
Out[5]:

```
<seaborn.axisgrid.PairGrid at 0x2925af8cc40>
```



```
In [6]: sns.displot(df['Avg. Area Income'])
```

```
Out[6]: <seaborn.axisgrid.FacetGrid at 0x2925727f880>
```



```
In [7]: df1=df.drop(['Address'],axis=1)
        df1
```

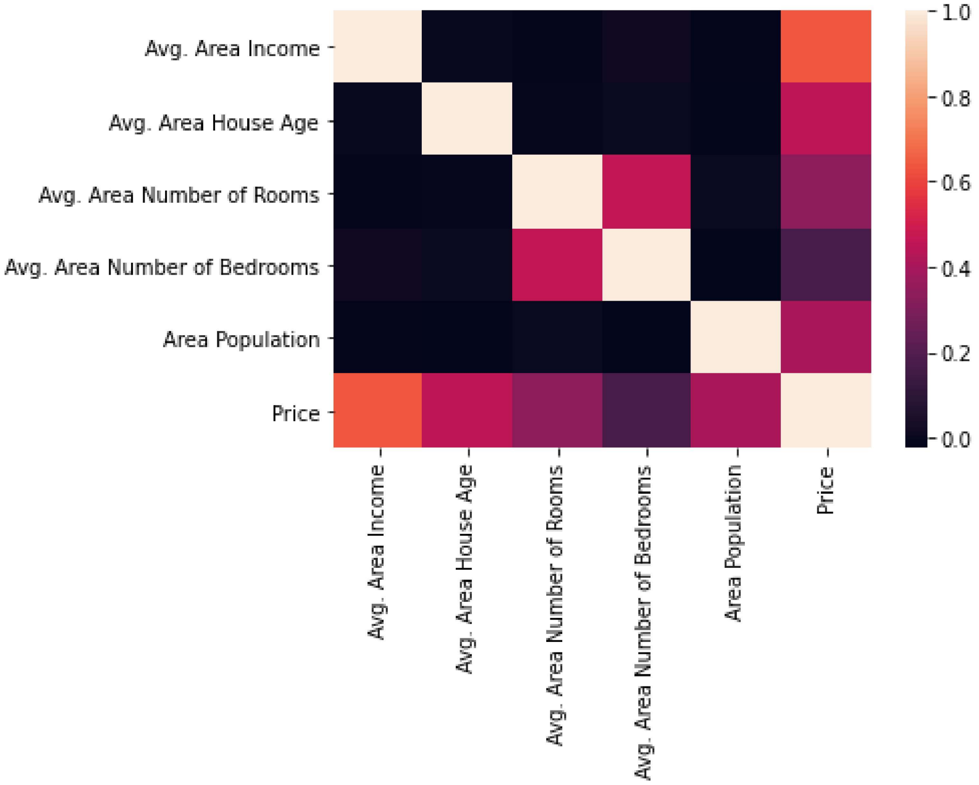
Out[7]:

	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
...	...	...	...	...	...	...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06

5000 rows × 6 columns

```
In [8]: sns.heatmap(df1.corr())
```

```
Out[8]: <AxesSubplot:>
```



```
In [9]: from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
```

In [11]:

```
y=df['Avg. Area Income']
x=df1.drop(['Avg. Area Income','Price'],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
print(x_train)
```

```
      Avg. Area House Age  Avg. Area Number of Rooms  \
4754             5.748358                   7.027627   
3297             5.653511                   6.690172   
1019             5.608290                   6.107279   
292              7.647151                   9.086808   
831              5.205825                   7.748964   
...                   ...                        ...   
4864             6.697178                   7.018083   
1965             7.722146                   5.791958   
2792             7.432324                   6.770586   
1112             5.379707                   7.309722   
2142             5.870558                   5.728607   

      Avg. Area Number of Bedrooms  Area Population  
4754                          3.29     33026.124336  
3297                          2.50     45828.215282  
1019                          2.28     48649.610385  
292                           3.21     33196.059837  
831                           4.33     43703.777582  
...                            ...              ...  
4864                          3.22     36950.451212  
1965                          4.33     64566.687380  
2792                          3.33     31225.123268  
1112                          6.25     41872.643278  
2142                          2.45     24511.819985  

[3500 rows x 4 columns]
```

In [12]:

```
model=LinearRegression()
model.fit(x_train,y_train)
model.intercept_
```

Out[12]:

```
69052.51662953071
```

In [13]:

```
coeff=pd.DataFrame(model.coef_,x.columns,columns=["Coefficient"])
coeff
```

Out[13]:

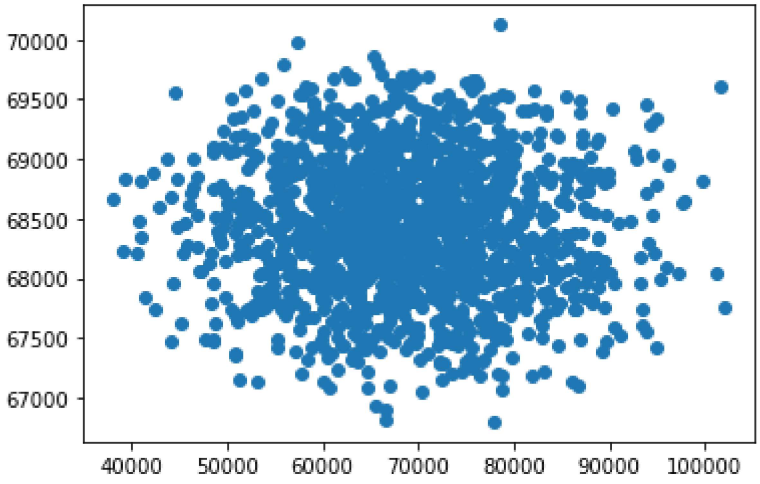
	Coefficient
Avg. Area House Age	194.762835
Avg. Area Number of Rooms	-385.067352
Avg. Area Number of Bedrooms	429.256596
Area Population	-0.021906

In [14]:

```
prediction=model.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[14]:

```
<matplotlib.collections.PathCollection at 0x2925fb545b0>
```



In [15]:

```
model.score(x_test,y_test)
```

Out[15]:

```
-0.0074141058159631346
```

In [16]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [17]:

```
rr = Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[17]:

```
Ridge(alpha=10)
```

In [18]:

```
rr.score(x_test,y_test)
```

Out[18]:

```
-0.0073878183105444695
```

In [19]:

```
la = Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[19]:

```
Lasso(alpha=10)
```

In [20]:

```
la.score(x_test,y_test)
```

Out[20]:

```
-0.007124242133762504
```

In [ ]: