Data Collection

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

In [11]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\usa_house.csv")
df
```

Out[11]:

	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 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\r AE 0
...	...	...	...	...	...	...	
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\r 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 G 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 [13]:
```
df.head(10)
```

Out[13]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Addres
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Ap 674\nLaurabury, N 3701
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson View Suite 079\nLak Kathleen, CA
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabe Stravenue\nDanieltow WI 06482
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO A 4482
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFP AE 0938
5	80175.754159	4.988408	6.104512	4.04	26748.428425	1.068138e+06	06039 Jennifer Islanc Apt. 443\nTracypor KS
6	64698.463428	6.025336	8.147760	3.41	60828.249085	1.502056e+06	4759 Daniel Shoa Sui 442\nNguyenburgh, C
7	78394.339278	6.989780	6.620478	2.42	36516.358972	1.573937e+06	972 Joyc Viaduct\nLake Williar TN 17778-648
8	59927.660813	5.362126	6.393121	2.30	29387.396003	7.988695e+05	USS Gilbert\nFPO A 2095
9	81885.927184	4.423672	8.167688	6.10	40149.965749	1.545155e+06	Unit 9446 Bc 0958\nDPO AE 9702

In [14]:
```
df.describe()
```

Out[14]:

	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 [15]:
```
df.info()
```

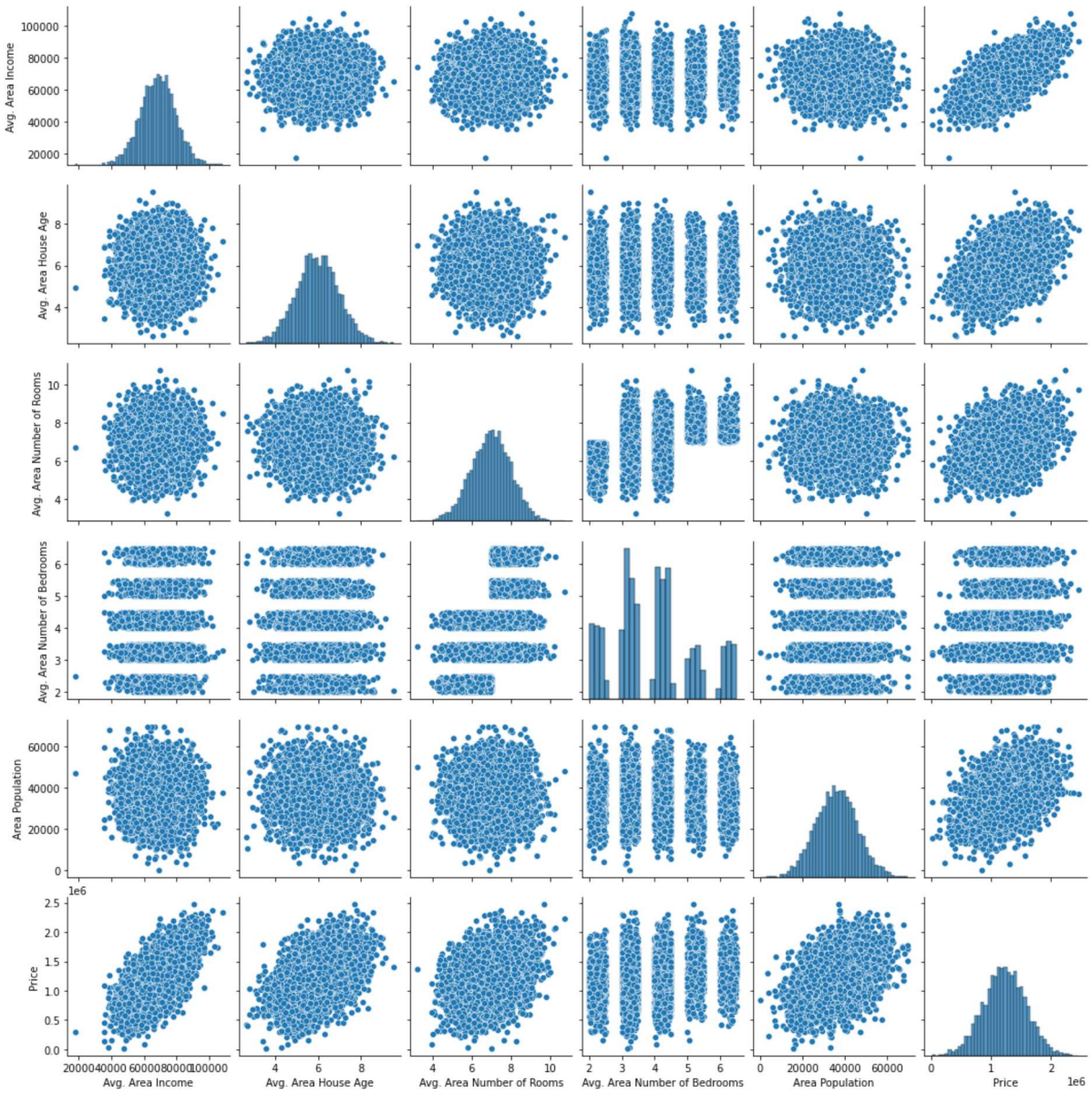
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   Avg. Area Income             5000 non-null   float64
 1   Avg. Area House 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 [17]:
```
df.columns
```

Out[17]:
```
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 [19]:
```
sns.pairplot(df)
```

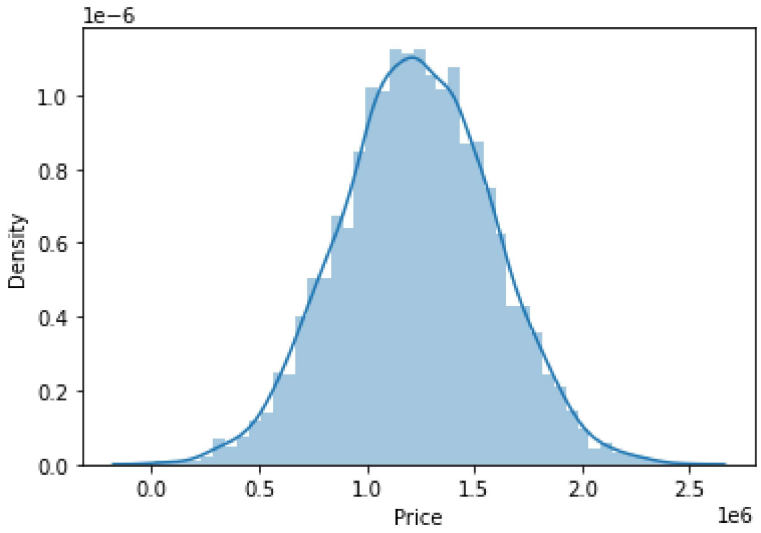
Out[19]:
```
<seaborn.axisgrid.PairGrid at 0x2b14393ba60>
```



In [23]:
```
sns.distplot(df['Price'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

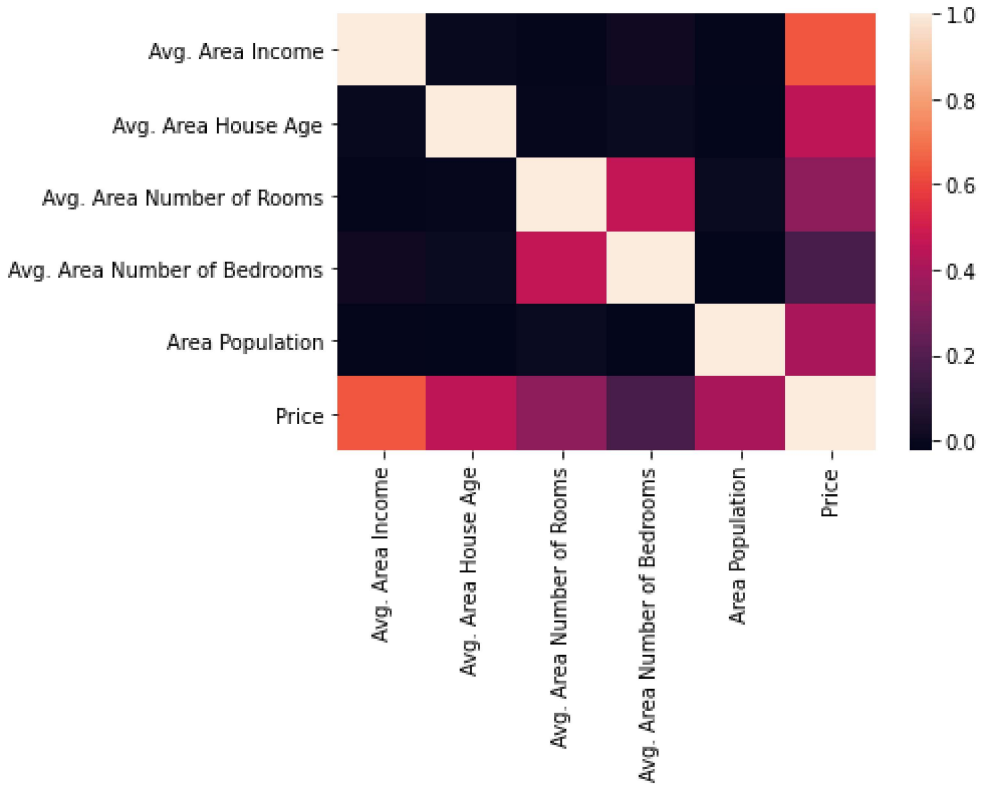
Out[23]:
```
<AxesSubplot:xlabel='Price', ylabel='Density'>
```



In [24]:
```
d=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address']]
```

In [26]:
```
sns.heatmap(d.corr())
```

Out[26]: <AxesSubplot:>



To TRAIN THE MODEL=MODEL BUILDING

WE ARE GOING TO TRAIN LINEAR REGRESSION MODEL;WE NEED TO SPLIT OUT DATA INTO TWO VARIABLES X AND Y IS INDEPENDENT VARIABLE (INPUT) AND Y IS DEPENDENT ON X (OUTPUT) WE COULD IGNORE ADDRESS COLUMN AS IT IS NOT REQUIRED FOR OUR MODEL

In [35]:
```
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 [33]:
```
#to split my dataset into traning and test data

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 [34]:
```
from sklearn.linear_model import LinearRegression

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

Out[34]: LinearRegression()

In [36]:
```
print(lr.intercept_)
```

```
-2643828.436924613
```

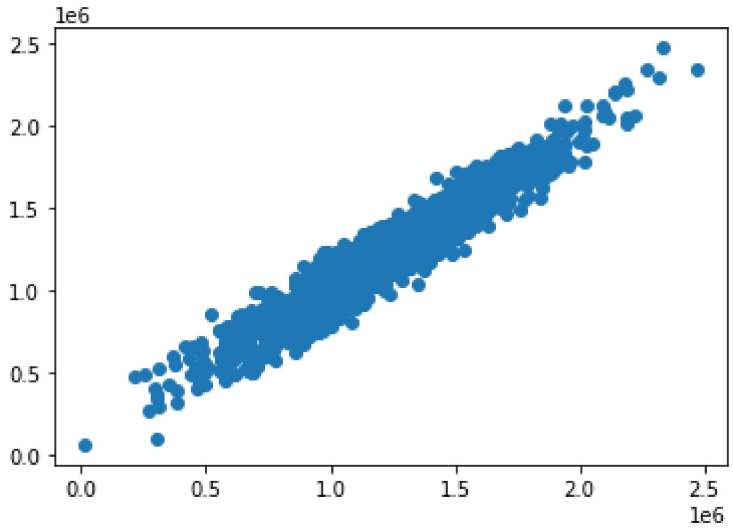
In [37]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['co-effecient'])
coeff
```

Out[37]:

	co-effecient
Avg. Area Income	21.656230
Avg. Area House Age	165883.890110
Avg. Area Number of Rooms	119674.300542
Avg. Area Number of Bedrooms	2401.719084
Area Population	15.264766

In [39]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[39]:
```
<matplotlib.collections.PathCollection at 0x2b148735370>
```



In [40]:
```
print(lr.score(x_test,y_test))
```

```
0.9192180487394439
```

In [ ]: