Problem Statement:

```
 A real estste agent wants help to predict the house price for regions
in USA. He gave you the dataset to work on and
 we decided to use Linear Regressionmodel. Creating a model that will
help to estimate of what the house would sell for
```

In [1]:
```
#importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
```

In [2]:
```
from sklearn.linear_model import LinearRegression
```

Data collection:

In [3]:
```
#Reading the data
df=pd.read_csv(r"C:/Users/my pc/downloads/USA_Housing.csv")
df
```

Out[3]:

	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\nLaurabu 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\nDanie WI 06
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFF 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 925{ 8489\nAPO AA 4
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy G Suite 076\nJoshu V/
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFF
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George F Apt. 509\nEast N

5000 rows × 7 columns

In [4]:
```
df.head()          #displays the first 5 rows
```

Out[4]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Addre
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry A 674\nLaurabury, N 3701
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Vie Suite 079\nLa 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 448
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFF AE 093

In [5]:
```
df.tail()          #displays the last 5 rows
```

Out[5]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFPO AP 30153-7653
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42991-3352
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garden Suite 076\nJoshualand, VA 01...
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO AE 73316
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridges Apt. 509\nEast Holly, NV 2...

In [6]:
```
df.describe()
```

Out[6]:

	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 [7]:
```
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 [8]:
```
df.isna().any()      #checking for null values
```

Out[8]:
```
Avg. Area Income                False
Avg. Area House Age             False
Avg. Area Number of Rooms       False
Avg. Area Number of Bedrooms    False
Area Population                 False
Price                           False
Address                         False
dtype: bool
```

In [9]:
```
df.shape
```

Out[9]:
```
(5000, 7)
```

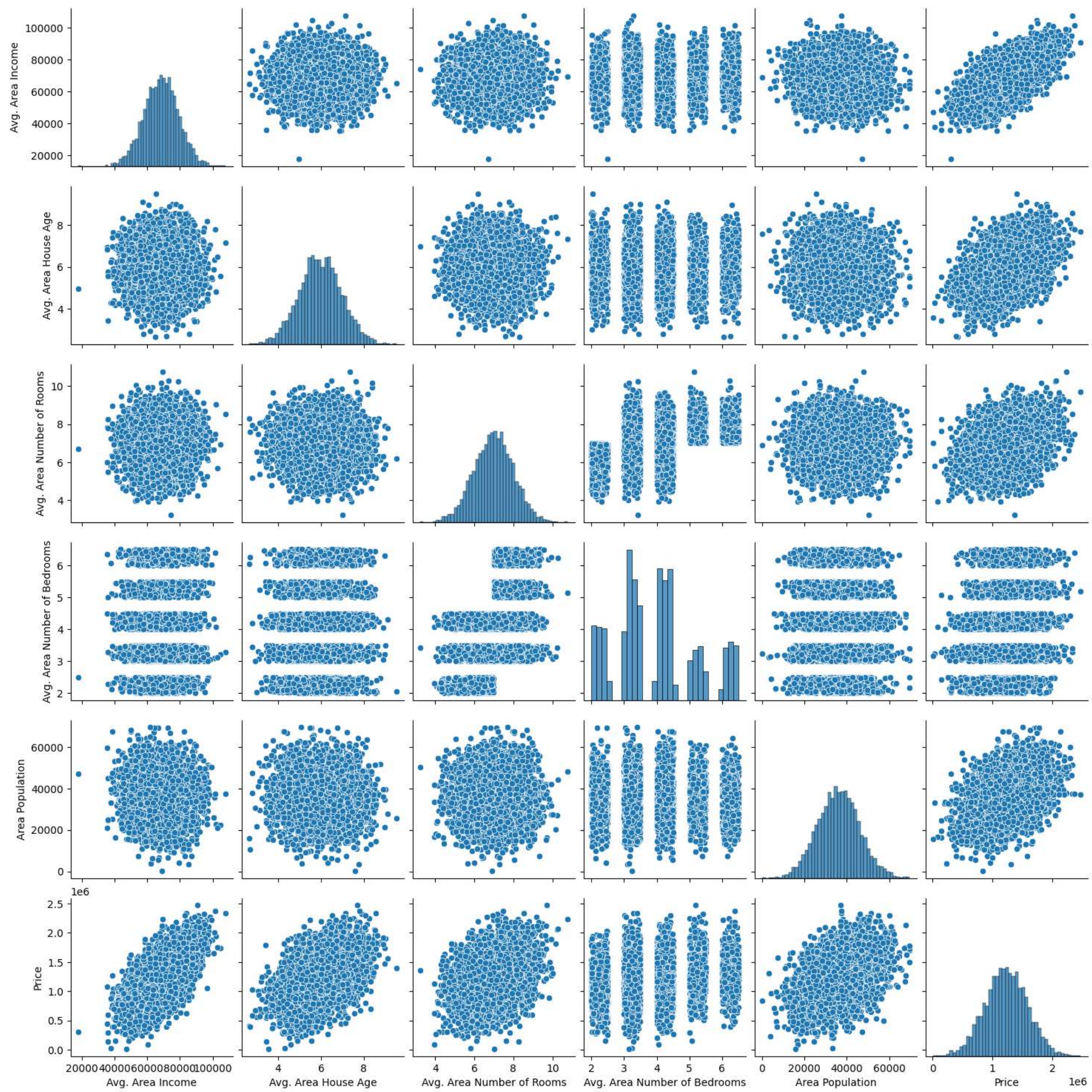
In [10]:
```
df.columns
```

Out[10]:
```
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')
```

Exploratory Data Analysis

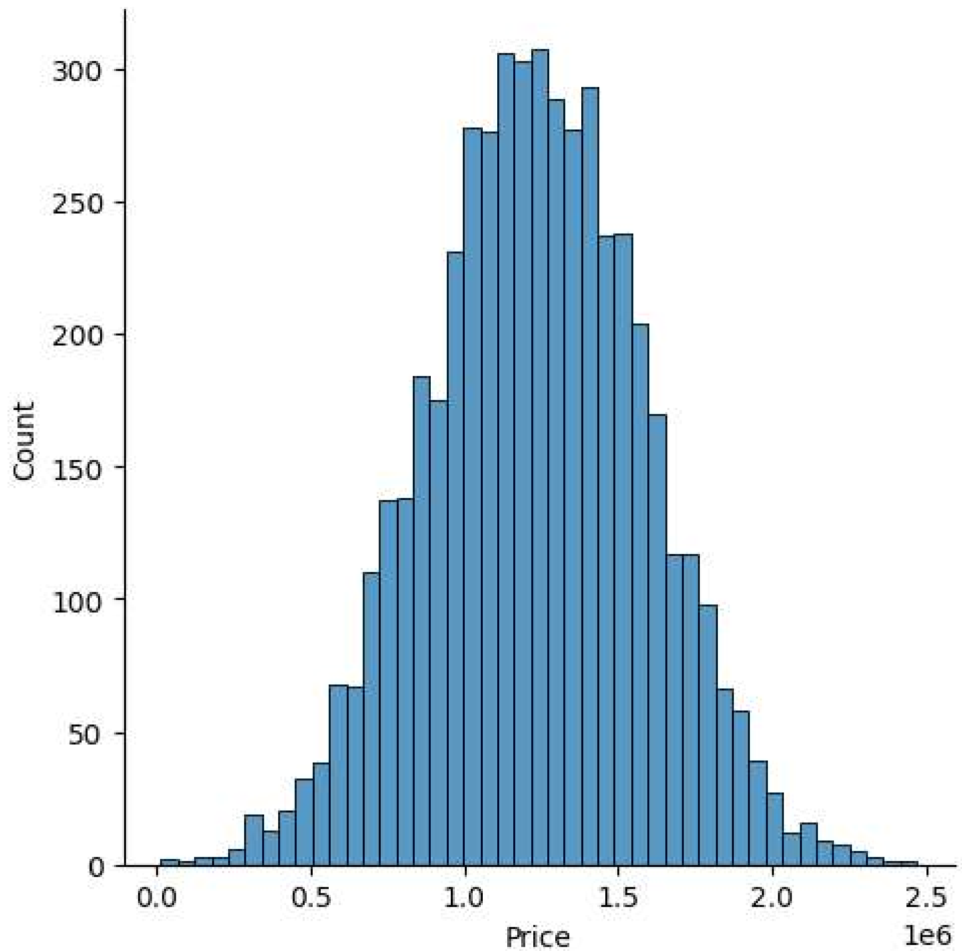
In [11]:
```
sns.pairplot(df)
```

Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x2767b386a70>
```



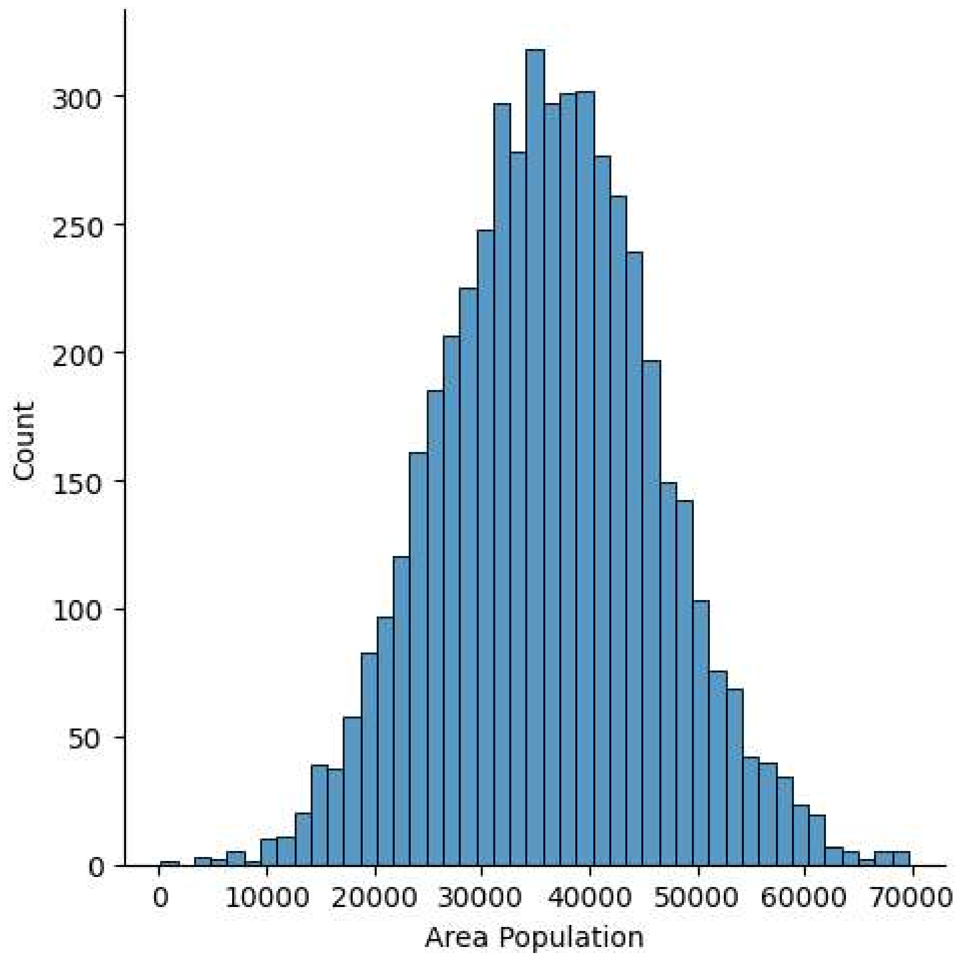
In [12]:
```
sns.displot(df['Price'])
```

Out[12]: <seaborn.axisgrid.FacetGrid at 0x2767aff6470>



In [13]:
```
sns.displot(df['Area Population'])
```

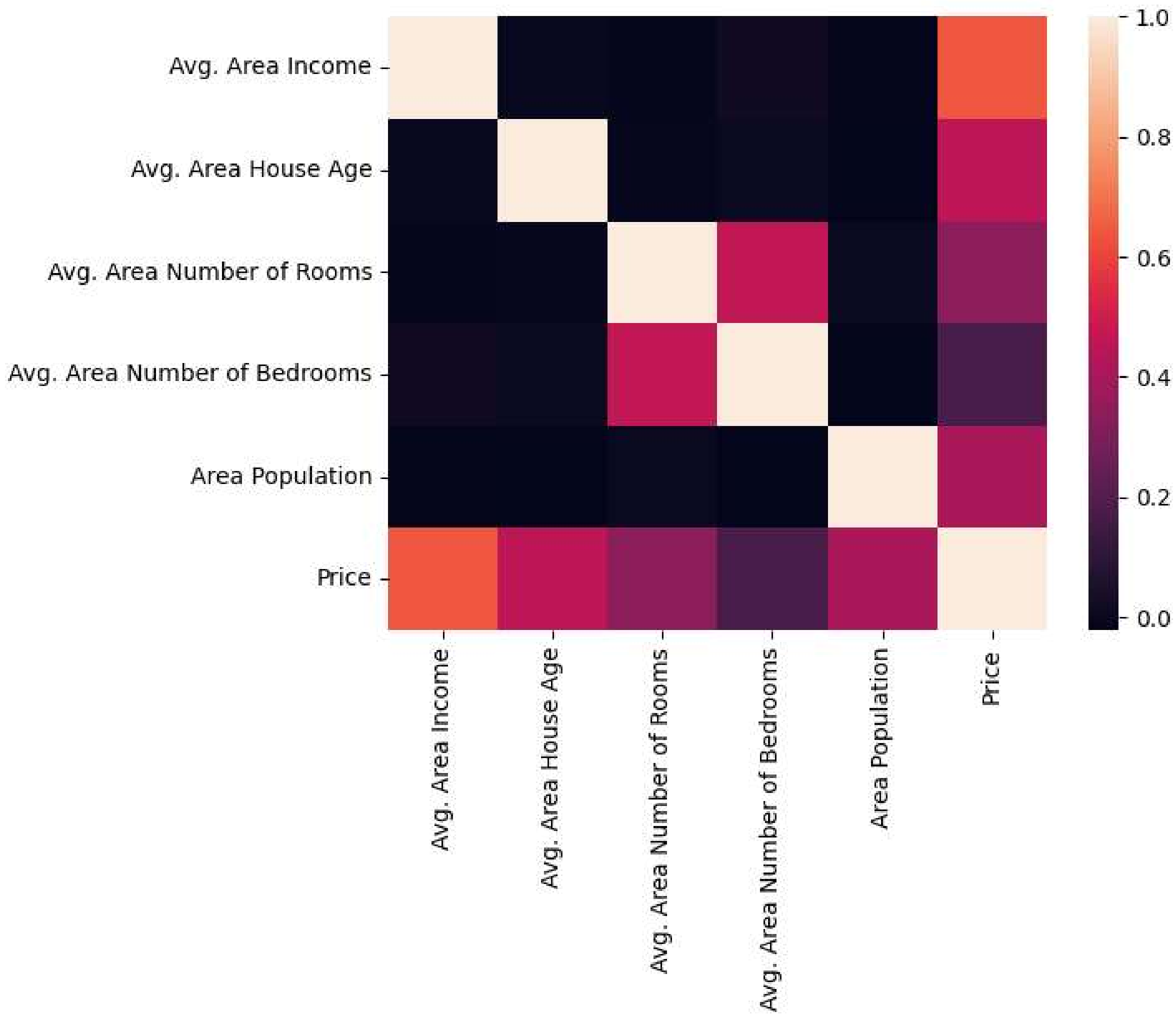
Out[13]: <seaborn.axisgrid.FacetGrid at 0x27669beaaa0>



In [14]:
```
Housedf=df[["Avg. Area Income","Avg. Area House Age","Avg. Area Number of Room
            "Area Population","Price"]]
```

In [15]:
```
sns.heatmap(Housedf.corr())
```

Out[15]: <Axes: >



To train the model

In [16]:
```
x=Housedf[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Roo
       'Avg. Area Number of Bedrooms', 'Area Population']]
y=df['Price']
```

In [17]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state
```

In [18]:
```
from sklearn.linear_model import LinearRegression
```

In [19]:
```
lm=LinearRegression()
lm.fit(x_train,y_train)
```

Out[19]:
```
▾ LinearRegression

LinearRegression()
```

In [20]:
```
print(lm.intercept_)
```

```
-2644850.0694553573
```

In [21]:
```
coeff_df=pd.DataFrame(lm.coef_,x.columns,columns=['coefficient'])
coeff_df
```

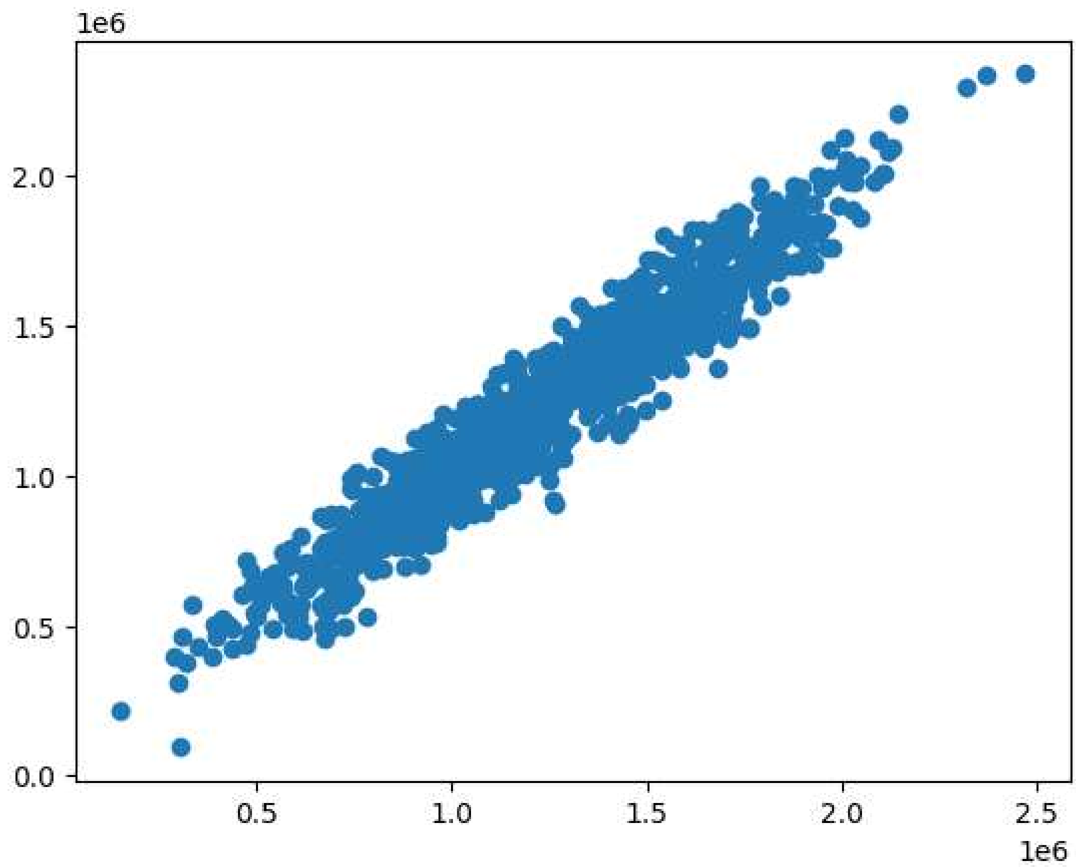
Out[21]:

	coefficient
Avg. Area Income	21.664598
Avg. Area House Age	165789.776062
Avg. Area Number of Rooms	120587.850072
Avg. Area Number of Bedrooms	1431.988439
Area Population	15.248314

Evaluating model performance

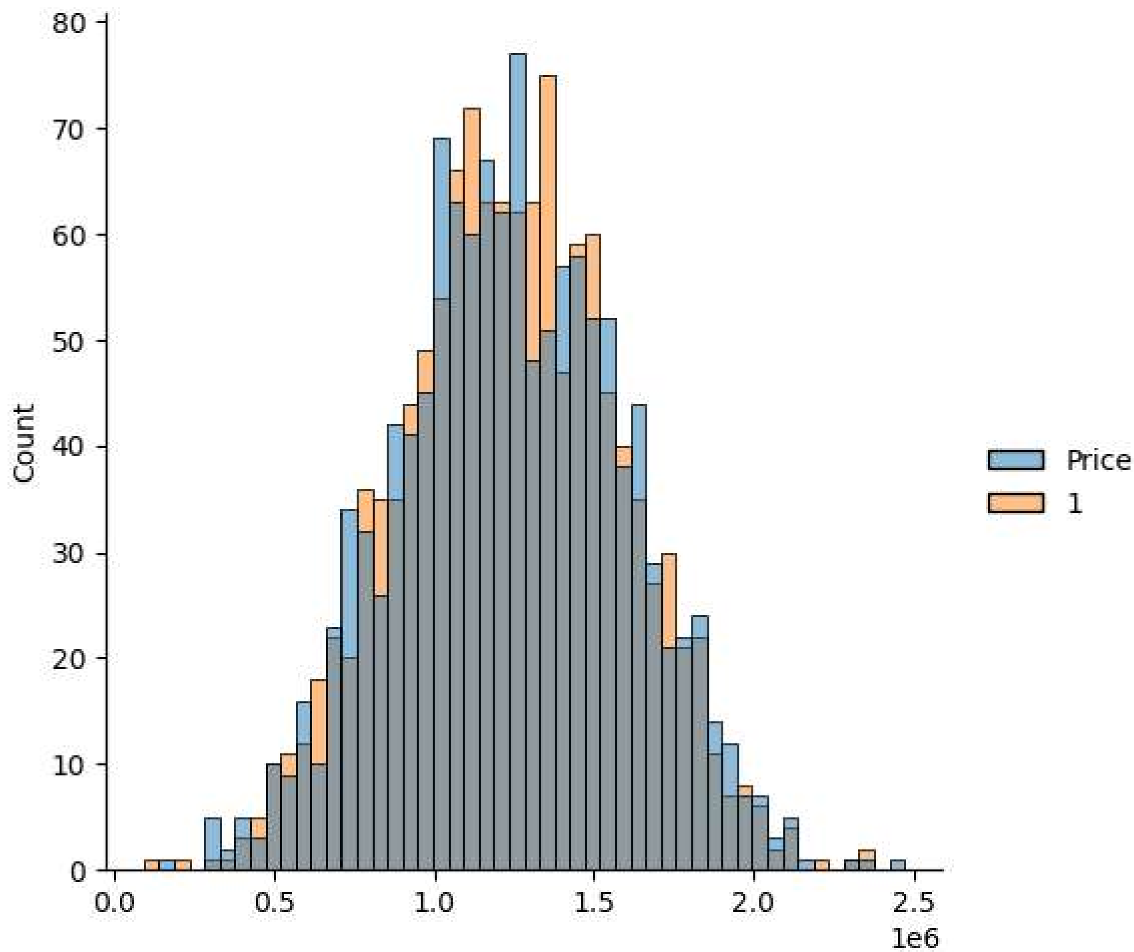
In [23]:
```
predictions=lm.predict(x_test)
plt.scatter(y_test,predictions)
```

Out[23]:
```
<matplotlib.collections.PathCollection at 0x2760120c3d0>
```



In [25]:
```
sns.displot((y_test,predictions),bins=50)
```

Out[25]: <seaborn.axisgrid.FacetGrid at 0x27600bc3580>



In [40]:
```
print("MAE:",metrics.mean_absolute_error(y_test,predictions))
print("MSE:",metrics.mean_squared_error(y_test,predictions))
print("RMSE:",np.sqrt(metrics.mean_squared_error(y_test,predictions)))
```

```
MAE: 81832.71748622792
MSE: 10401275405.858944
RMSE: 101986.64327184684
```

In [38]:
```
from sklearn.metrics import r2_score
r2=r2_score(y_test,predictions)
print("R2 score:",r2)
```

```
R2 score: 0.918508746677805
```

conclusion:

It is best model ,because the r2 is high and the linear and normal distribution are best fitted for this dataset