In [1]:

```
#importing the required packages
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline
```

In [2]:

```
#importing and viewing dataset
HouseDF = pd.read_csv('USA_Housing.csv')
HouseDF.head()
```

Out[2]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Addr
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, C
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizal Stravenue\nDanielto WI 064
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO 44
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nF AE 09

In [3]:

```
HouseDF.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]:

```
HouseDF.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]:

```
HouseDF.columns
```

Out[5]:

```
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 [6]:

```
HouseDF.corr()
```

Out[6]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
Avg. Area Income	1.000000	-0.002007	-0.011032	0.019788	-0.016234	0.639734
Avg. Area House Age	-0.002007	1.000000	-0.009428	0.006149	-0.018743	0.452543
Avg. Area Number of Rooms	-0.011032	-0.009428	1.000000	0.462695	0.002040	0.335664
Avg. Area Number of Bedrooms	0.019788	0.006149	0.462695	1.000000	-0.022168	0.171071
Area Population	-0.016234	-0.018743	0.002040	-0.022168	1.000000	0.408556
Price	0.639734	0.452543	0.335664	0.171071	0.408556	1.000000

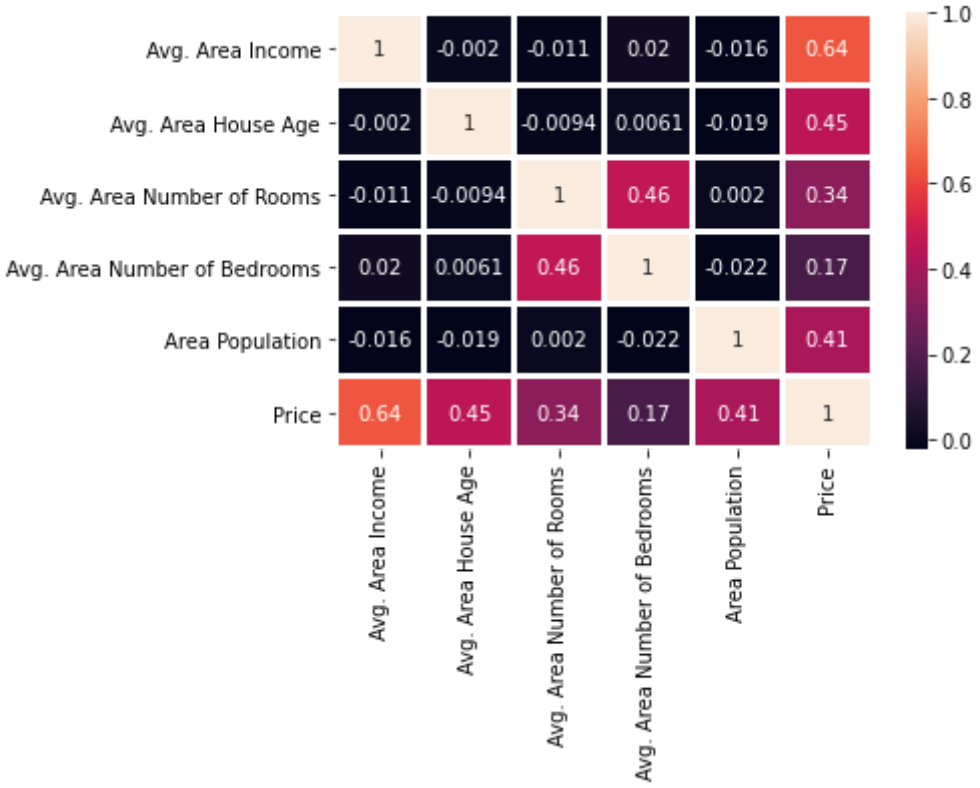
In [7]:

```
sns.heatmap(HouseDF.corr(), annot=True, Linewidth = 2)
```

```
C:\Users\Ilakiya\anaconda3\lib\site-packages\seaborn\matrix.py:308: Matplotl
ibDeprecationWarning: Case-insensitive properties were deprecated in 3.3 and
support will be removed two minor releases later
  mesh = ax.pcolormesh(self.plot_data, cmap=self.cmap, **kws)
```

Out[7]:

```
<AxesSubplot:>
```

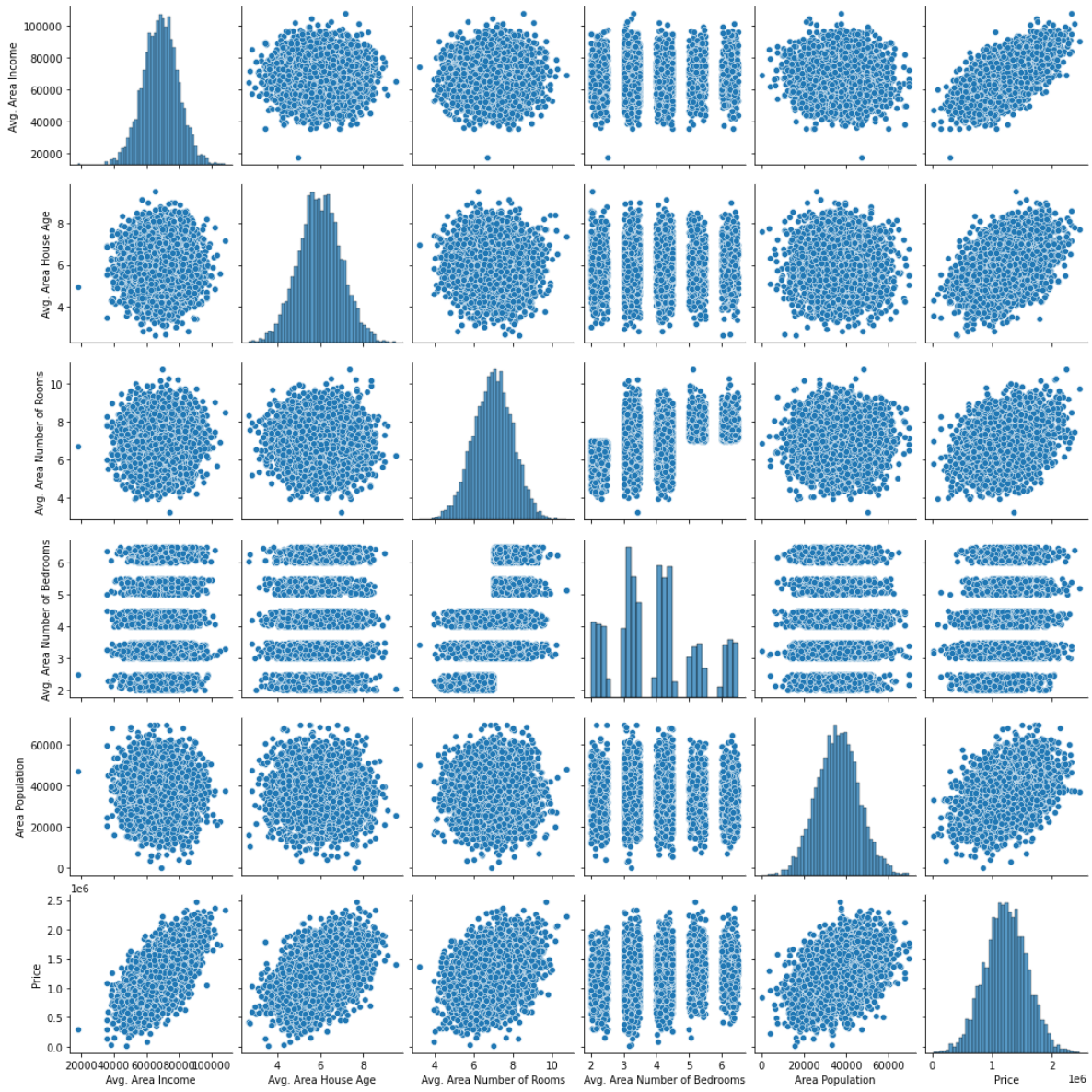


In [8]:

```
sns.pairplot(HouseDF)
```

Out[8]:

```
<seaborn.axisgrid.PairGrid at 0x1e223d1d400>
```



In [11]:

```
X = HouseDF[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
             'Avg. Area Number of Bedrooms', 'Area Population']]

Y= HouseDF['Price']
```

In [12]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4, random_state=43)
```

In [13]:

```
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(3000, 5)
(2000, 5)
(3000,)
(2000,)
```

In [14]:

```
from sklearn.linear_model import LinearRegression

lm = LinearRegression()
lm.fit(X_train,Y_train)
```

Out[14]:

```
LinearRegression()
```

In [15]:

```
print(lm.intercept_)
```

```
-2627588.9438994704
```

In [16]:

```
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
```

In [17]:

```
coeff_df
```

Out[17]:

	Coefficient
Avg. Area Income	21.393705
Avg. Area House Age	166749.064003
Avg. Area Number of Rooms	120829.501497
Avg. Area Number of Bedrooms	1199.374695
Area Population	15.112633

In [18]:

```
#Holding all other features fixed, a 1 unit increase in Avg. Area Income is associated with
#Holding all other features fixed, a 1 unit increase in Avg. Area House Age is associated w
#Holding all other features fixed, a 1 unit increase in Avg. Area Number of Rooms is associ
#Holding all other features fixed, a 1 unit increase in Avg. Area Number of Bedrooms is ass
#Holding all other features fixed, a 1 unit increase in Area Population is associated with
```

In [22]:

```
predictions = lm.predict(X_test)
print(predictions)
```

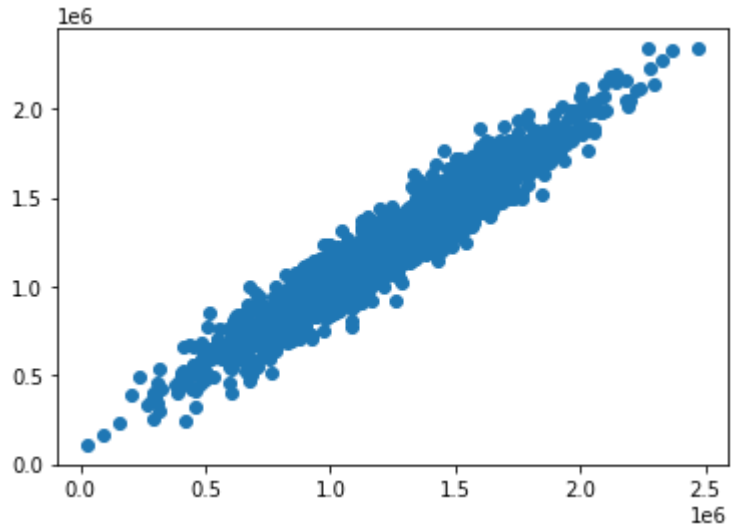
```
[ 715948.57982857 1573733.18738268 1149395.67252801 ...  754762.14560837
  865576.01768423 1476967.5348202 ]
```

In [25]:

```
plt.scatter(Y_test,predictions)
```

Out[25]:

```
<matplotlib.collections.PathCollection at 0x1e2272ac400>
```



In [28]:

```
#calculation of r2square
from sklearn.metrics import r2_score
r2_score(Y_test,predictions)
```

Out[28]:

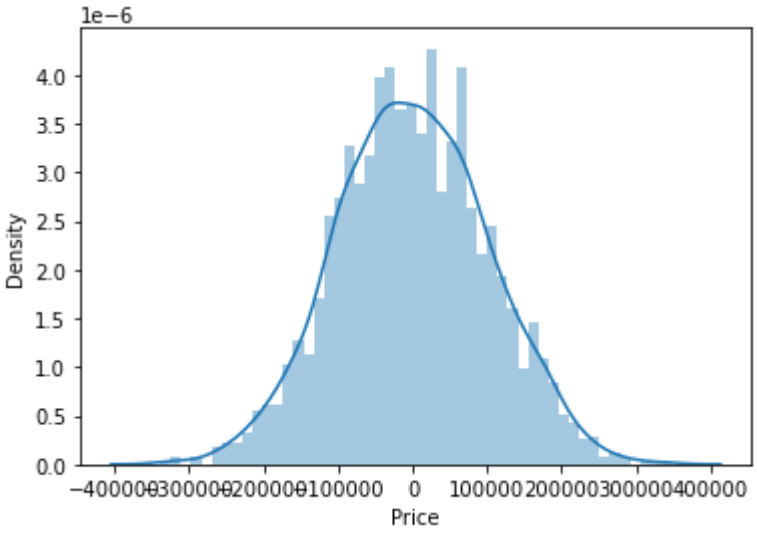
```
0.9192378242759394
```

from the scatter plot, we see data is in a line form, which means our model has done good predictions.

In [30]:

```
sns.distplot((Y_test-predictions),bins=50);
```

```
C:\Users\Ilakiya\anaconda3\lib\site-packages\seaborn\distributions.py:2551:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-lev
el function with similar flexibility) or `histplot` (an axes-level function
for histograms).
  warnings.warn(msg, FutureWarning)
```



from the above histogram plot, we see data is in bell shape (Normally Distributed), which means our model has done good predictions.

In [32]:

```
from sklearn import metrics

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: 81396.74372224537
MSE: 10262306887.381538
RMSE: 101303.04480804878
```

We have created a Linear Regression Model which we help the real state agent for estimating the house price.

In [ ]:

```

```