Import the required libraries

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

In [2]:
```
%matplotlib inline
```

Read data using Pandas

In [3]:
```
df = pd.read_csv('USA_Housing.csv')
```

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

Out[4]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Apt 674\nLaurabury, NE 3701..
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA..
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown, WI 06482..
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO AP 4482(
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO AE 0938(

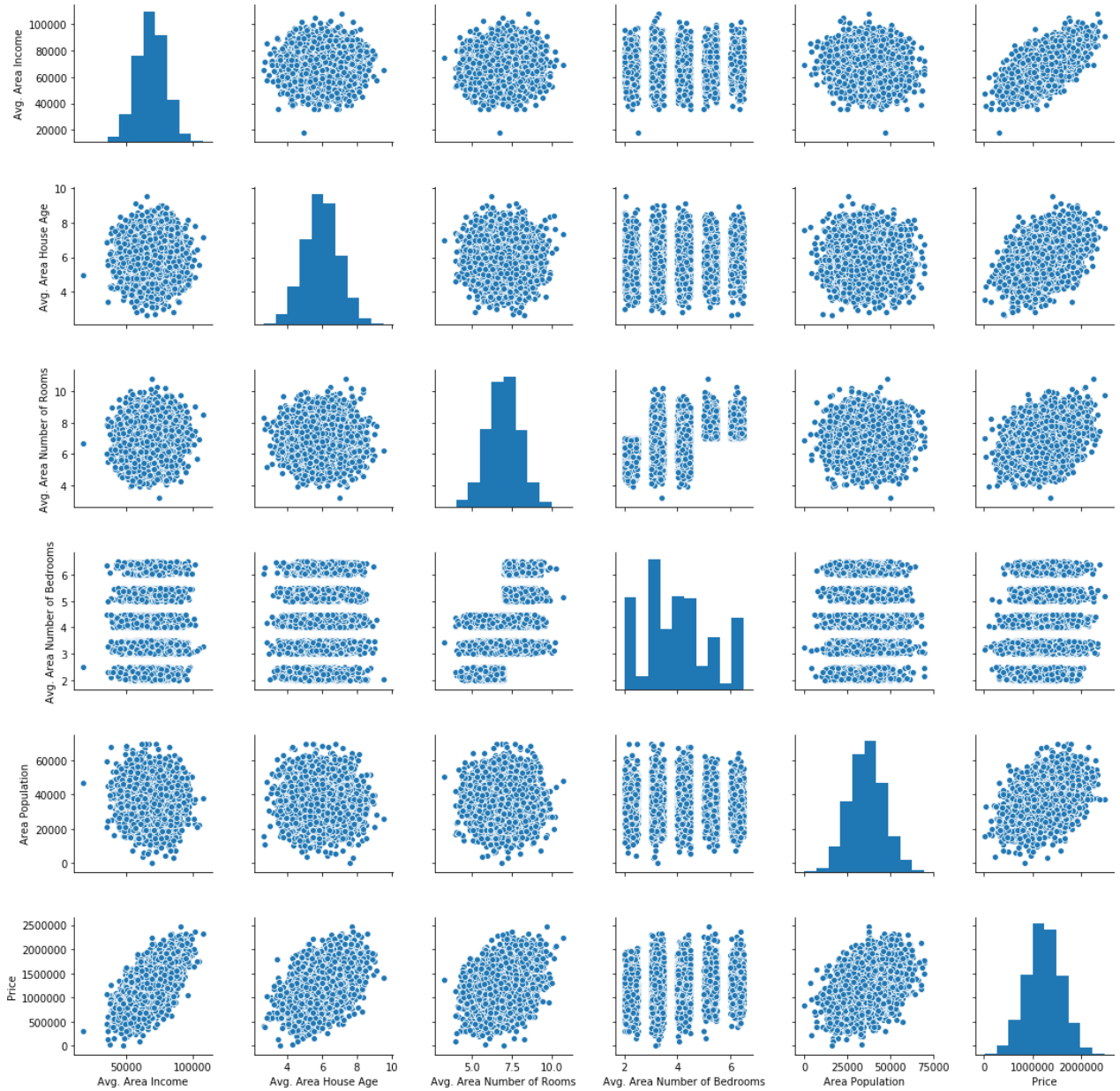
In [5]:
```
df.info()
```

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

Visualize data using Seaborn

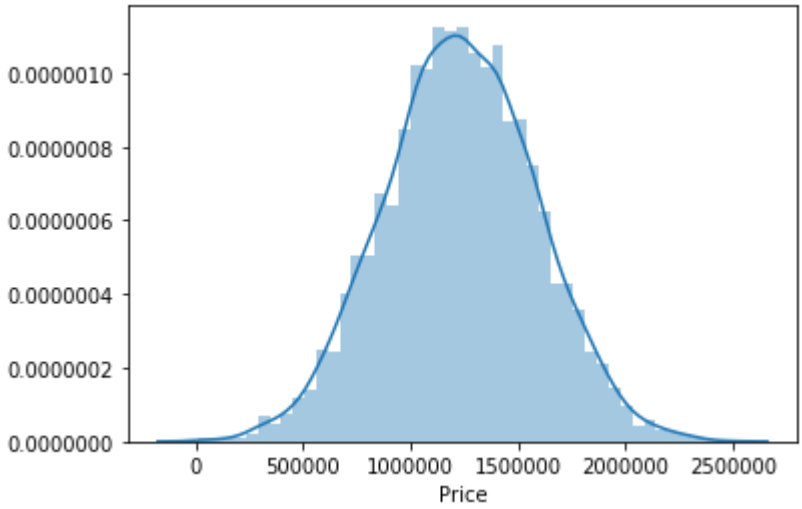
In [6]:
```
sns.pairplot(df)
```

Out[6]:
```
<seaborn.axisgrid.PairGrid at 0x1ead42c35c8>
```



In [7]:
```
sns.distplot(df['Price'])
```

Out[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1eadd4cd588>
```



In [8]:
```
df.corr()
```

Out[8]:

	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

Do Correlation Plotting

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

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1eae0bc68c8>
```

In [10]:
```
sns.heatmap(df.corr(), annot=True)
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1eae0c5cb48>
```

In [11]:
```
df.columns
```

Out[11]:
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],
      dtype='object')
```

In [12]:
```
X = df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price']]
```

In [13]:
```
y = df['Price']
```

Split Data using Skit-Learn

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=1
```

In [16]:
```
from sklearn.linear_model import LinearRegression
```

In [17]:
```
lm = LinearRegression()
```

Train Linear Regression model

In [19]:
```
lm.fit(X_train, y_train)
```

Out[19]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

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

```
[ 6.71985063e-16 -1.51241686e-10 -5.65370457e-11 -2.92687443e-12
 -7.69295651e-15  1.00000000e+00]
```

In [21]:
```
print(lm.intercept_)
```

```
1.6298145055770874e-09
```

In [22]:
```
X_train.columns
```

Out[22]:
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price'],
      dtype='object')
```

Create Coeff DataFrame

In [23]:
```
cdf = pd.DataFrame(lm.coef_, X.columns, columns = ['Coeff'])
```

In [24]:
```
cdf
```

Out[24]:

	Coeff
Avg. Area Income	6.719851e-16
Avg. Area House Age	-1.512417e-10
Avg. Area Number of Rooms	-5.653705e-11
Avg. Area Number of Bedrooms	-2.926874e-12
Area Population	-7.692957e-15
Price	1.000000e+00

Predict Data

In [25]:
```
predictions = lm.predict(X_test)
```

In [26]:
```
predictions
```

Out[26]:
```
array([1251688.61570287,  873048.31964236, 1696977.6628326 , ...,
       151527.08262656, 1343824.21514432, 1906024.63648501])
```

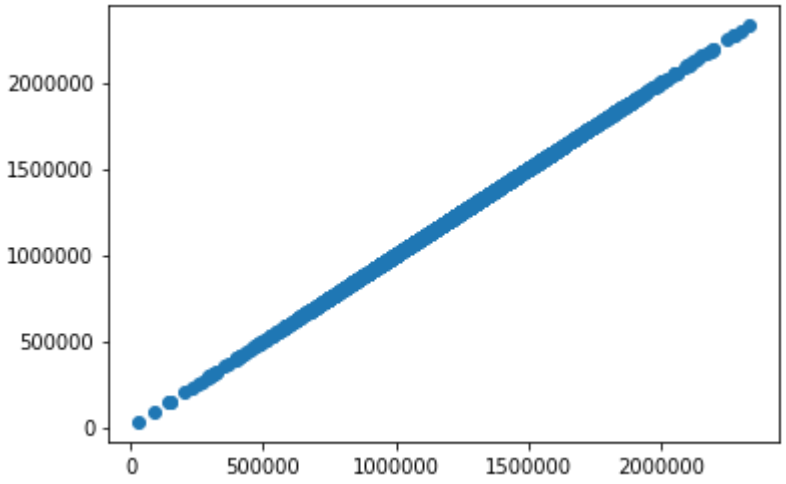
In [27]:
```
y_test
```

Out[27]:
```
1718    1.251689e+06
2511    8.730483e+05
345     1.696978e+06
2521    1.063964e+06
54      9.487883e+05
            ...
1776    1.489520e+06
4269    7.777336e+05
1661    1.515271e+05
2410    1.343824e+06
2302    1.906025e+06
Name: Price, Length: 2000, dtype: float64
```

Visualize predictions using matplotlib

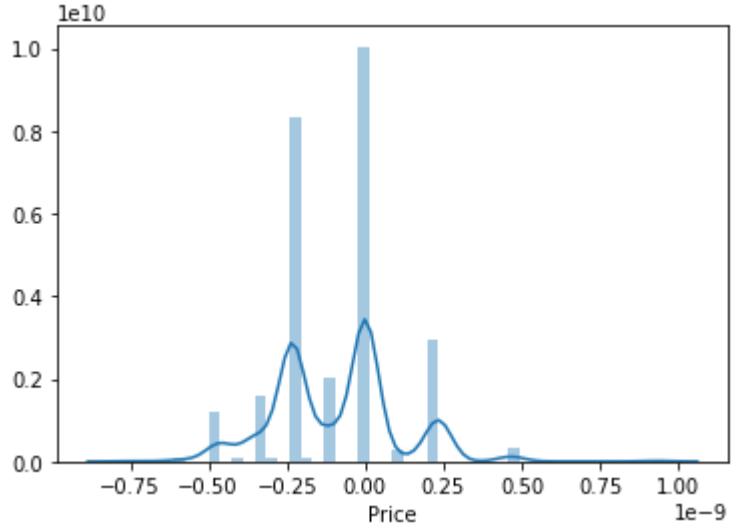
In [28]:
```
plt.scatter(y_test, predictions)
```

Out[28]:
```
<matplotlib.collections.PathCollection at 0x1eae1398288>
```



In [29]:
```
sns.distplot((y_test-predictions))
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1eae13c31c8>
```



Determine model evaluation parameters

In [30]:
```
from sklearn import metrics
```

In [31]:
```
metrics.mean_absolute_error(y_test, predictions)
```

Out[31]: 1.6116609913296998e-10

In [32]:
```
metrics.mean_squared_error(y_test, predictions)
```

Out[32]: 4.760497217136579e-20

In [33]:
```
np.sqrt(metrics.mean_squared_error(y_test, predictions))
```

Out[33]: 2.1818563695020298e-10

In [ ]:
```

```