Import Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

importing the data set

```
HouseDF = pd.read_csv('USA_Housing.csv')
```

```
HouseDF.head()
```

	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 Fer 674\nLaurabu 3
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Suite 079\ 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	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond' AE

```
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
```

```
HouseDF.describe()
```

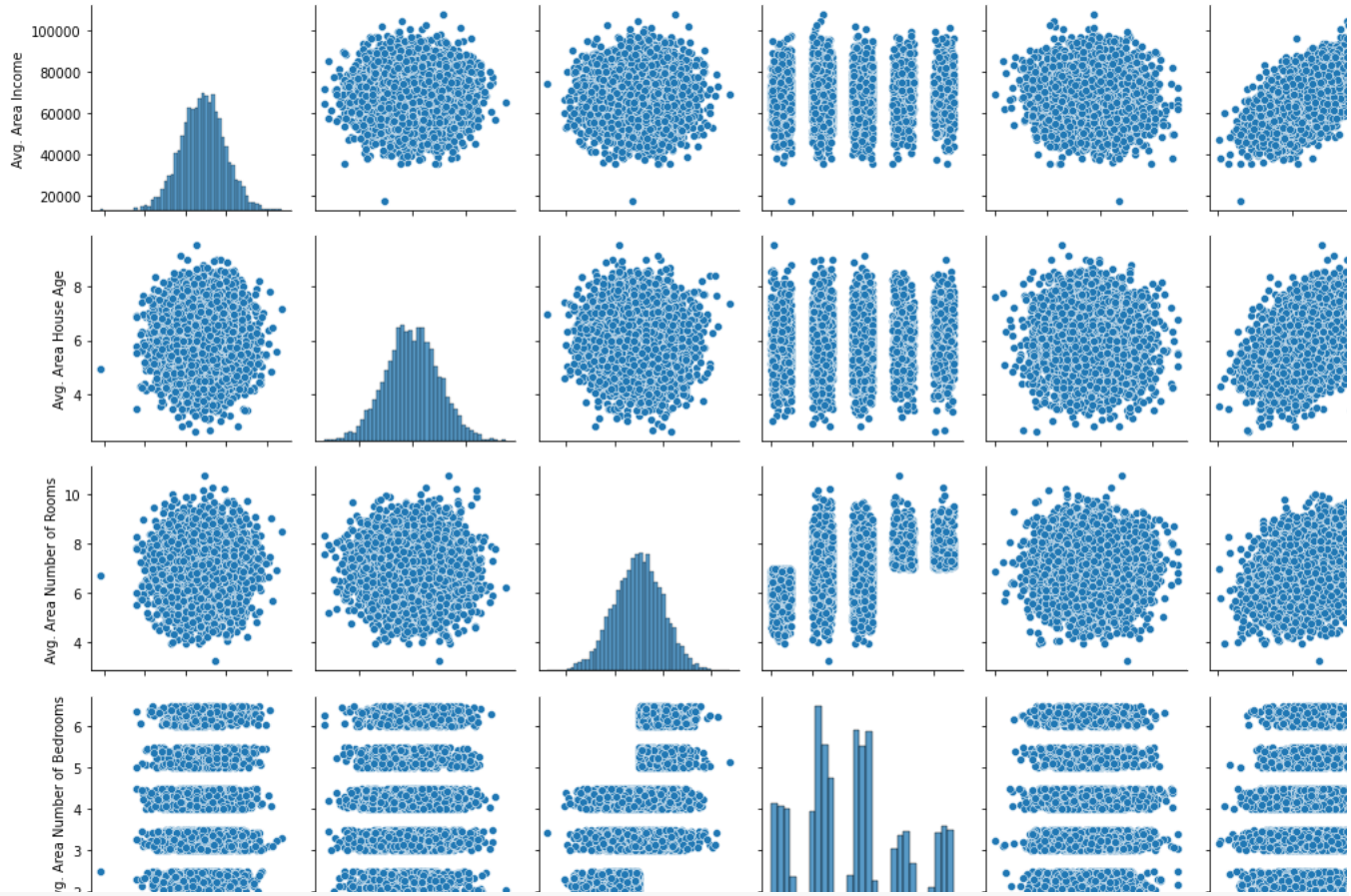
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Pr
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.000000e
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.232073e
std	10657.991214	0.991456	1.005833	1.234137	9925.650114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

```
HouseDF.columns
```

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

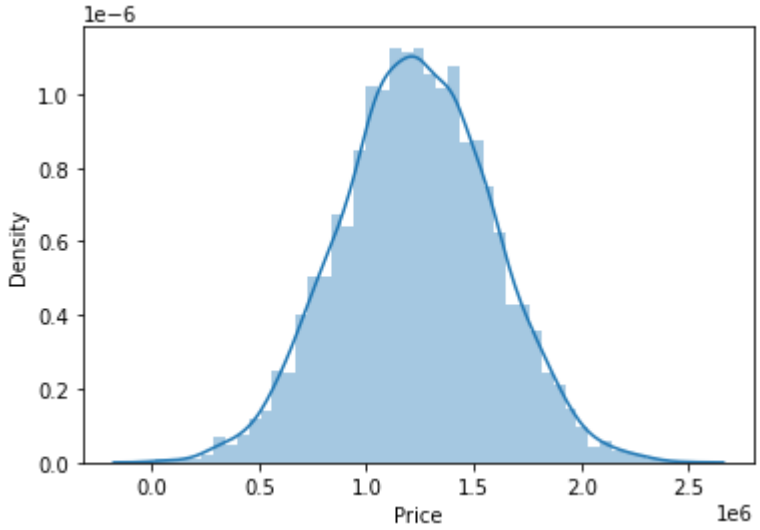
```
sns.pairplot(HouseDF)
```

```
<seaborn.axisgrid.PairGrid at 0x7f72626a5410>
```




```
sns.distplot(HouseDF['Price'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f725ed20350>
```




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

y = HouseDF['Price']
```

```
# Splitting the dataset into the Training set and Test set
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=101)
```

```
# Step2 - Fitting Simple Linear Regression to the Traning Set
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train,y_train)
```

```
LinearRegression()
```

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

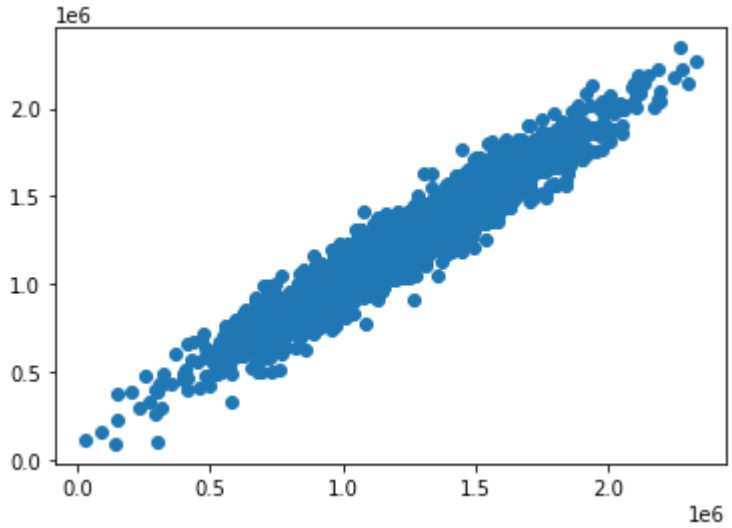
```
-2640159.7968526958
```

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

	Coefficient
Avg. Area Income	21.528276
Avg. Area House Age	164883.282027
Avg. Area Number of Rooms	122368.678027
Avg. Area Number of Bedrooms	2233.801864
Area Population	15.150420

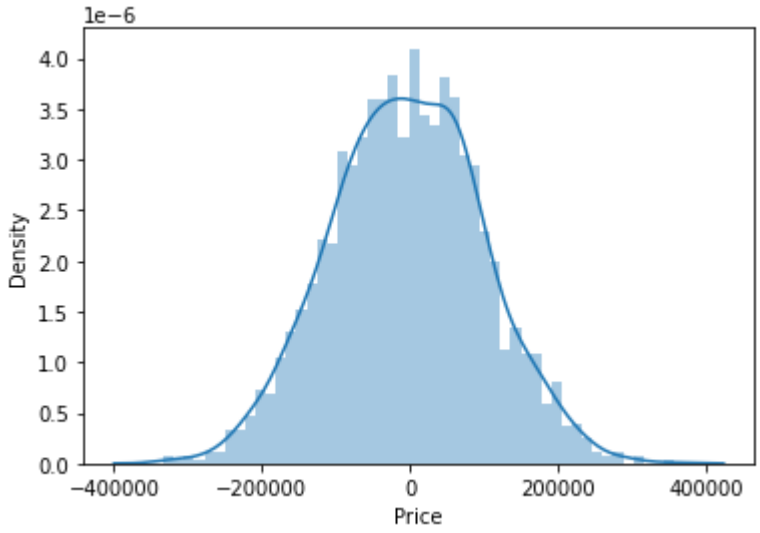
```
predictions = lm.predict(X_test)
plt.scatter(y_test,predictions)
```

```
<matplotlib.collections.PathCollection at 0x7f725a274e90>
```



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

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
```




```
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: 82288.22251914942
MSE: 10460958907.20898
RMSE: 102278.82922290899
```

```

```