

Problem Statement:

A real estate agent want to help to predict the house price for regions in USA.He gave us the dataset to work on to use Linear Regression modelCreate a Model that helps him to estimate of what the house would sell for

```
In [2]: #import libraries  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns
```

```
In [3]: #import dataset
df=pd.read_csv(r"E:\154\10_USA_Housing - 10_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	Address
0	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06	208 Michael Ferry / 674\nLaurabury, 370
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06	188 Johnson Vie Suite 079\nLi Kathleen, C
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06	9127 Elizab Stravenue\nDanielto WI 0648
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06	USS Barnett\nFPO 44i
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05	USNS Raymond\nF AE 09:
...
4995	60567.94414	7.830362	6.137356	3.46	22837.36103	1.060194e+06	USNS Williams\nF AP 30153-7i
4996	78491.27543	6.999135	6.576763	4.02	25616.11549	1.482618e+06	PSC 9258, I 8489\nAPO AA 429 3:
4997	63390.68689	7.250591	4.805081	2.13	33266.14549	1.030730e+06	4215 Tracy Gar Suite 076\nJoshuala VA C
4998	68001.33124	5.534388	7.130144	5.44	42625.62016	1.198657e+06	USS Wallace\nFPO 73:
4999	65510.58180	5.992305	6.792336	4.07	46501.28380	1.298950e+06	37778 George Rid Apt. 509\nEast H NV

5000 rows × 7 columns



In [4]: `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 [4]: `#to display top 5 rows`
`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	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA...
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05	USNS Raymond\nFPO AE 09386

Data cleaning and Pre-Processing

In [5]: *#To find null values*
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 [6]: *# To display summary of statistics*
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%	61480.562390	5.322283	6.299250	3.140000	29403.928700	9.975771e+05
50%	68804.286405	5.970429	7.002902	4.050000	36199.406690	1.232669e+06
75%	75783.338665	6.650808	7.665871	4.490000	42861.290770	1.471210e+06
max	107701.748400	9.519088	10.759588	6.500000	69621.713380	2.469066e+06

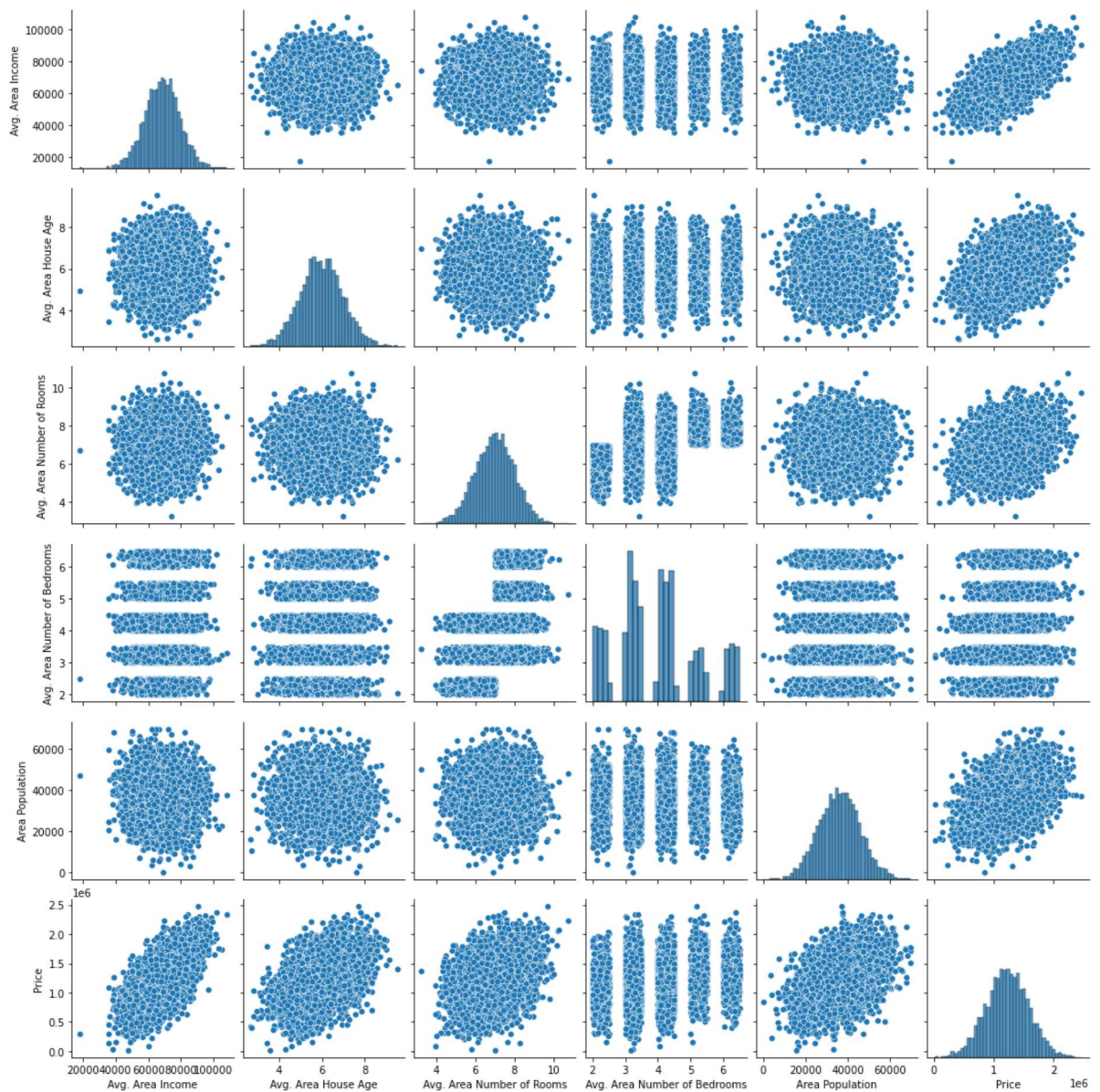
In [7]: *#To Display column heading*
df.columns

Out[7]: 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')

EDA and VISUALIZATION

```
In [8]: sns.pairplot(df)
```

```
Out[8]: <seaborn.axisgrid.PairGrid at 0x28e6d2a2be0>
```

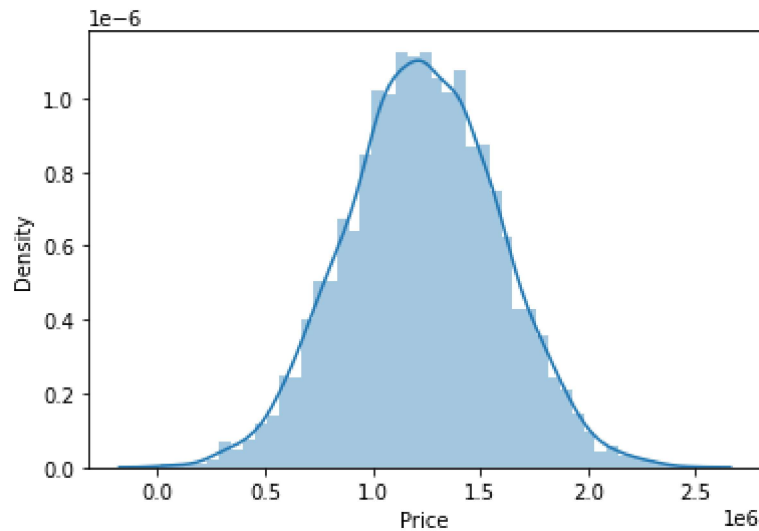


```
In [9]: sns.distplot(df[""])
```

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

```
warnings.warn(msg, FutureWarning)
```

```
Out[9]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



```
In [10]: df1=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',  
                'Avg. Area Number of Bedrooms', 'Area Population', 'Price']]
```

Plot Using Heat Map

```
In [11]: sns.heatmap(df1.corr())
```

```
Out[11]: <AxesSubplot:>
```



To Train The Model-Model Building

we are going to train Linera Regression Model;We need to split out data into two variables x and y where x is independent variable(input) and y is dependent on x(output) we could ignore address column as it required for our model

```
In [12]: x=df1[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
               'Avg. Area Number of Bedrooms', 'Area Population']]
          y=df1['Price']
```

To Split my dataset into training and test data

```
In [13]: 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 [17]: from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

Out[17]: LinearRegression()

```
In [19]: lr.intercept_
```

Out[19]: -2659434.4926807457

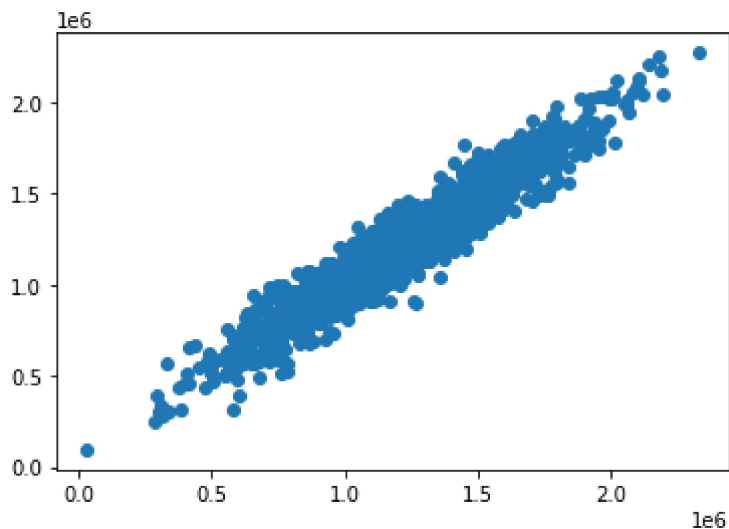
```
In [22]: coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[22]:

	Co-efficient
Avg. Area Income	21.844909
Avg. Area House Age	165514.444131
Avg. Area Number of Rooms	121080.002332
Avg. Area Number of Bedrooms	2546.084156
Area Population	15.144061

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

Out[23]: <matplotlib.collections.PathCollection at 0x28e75aa3d30>



```
In [24]: lr.score(x_test,y_test)
```

Out[24]: 0.9133849291100872

```
In [ ]:
```