Linerar Regression Model For House Price Prediction

April 7, 2024

0.0.1 Linear Regression Machine Learning Project for House Price Prediction - Prodigy Infotech ML Project - 1

0.0.2 Import Libraries

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

0.0.3 Importing Data and Checking out.

[7]:
```
df = pd.read_csv(r"D:\Software\New Project\Internship\Prodigy Infotech\House␣
↪Prediction - Linear Regression\USA_Housing.csv")
```

[8]:
```
df.head()
```

[8]:

	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 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO AE 09386

[9]:
```
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
```

[10]:
```
df.describe()
```

[10]:

	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

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

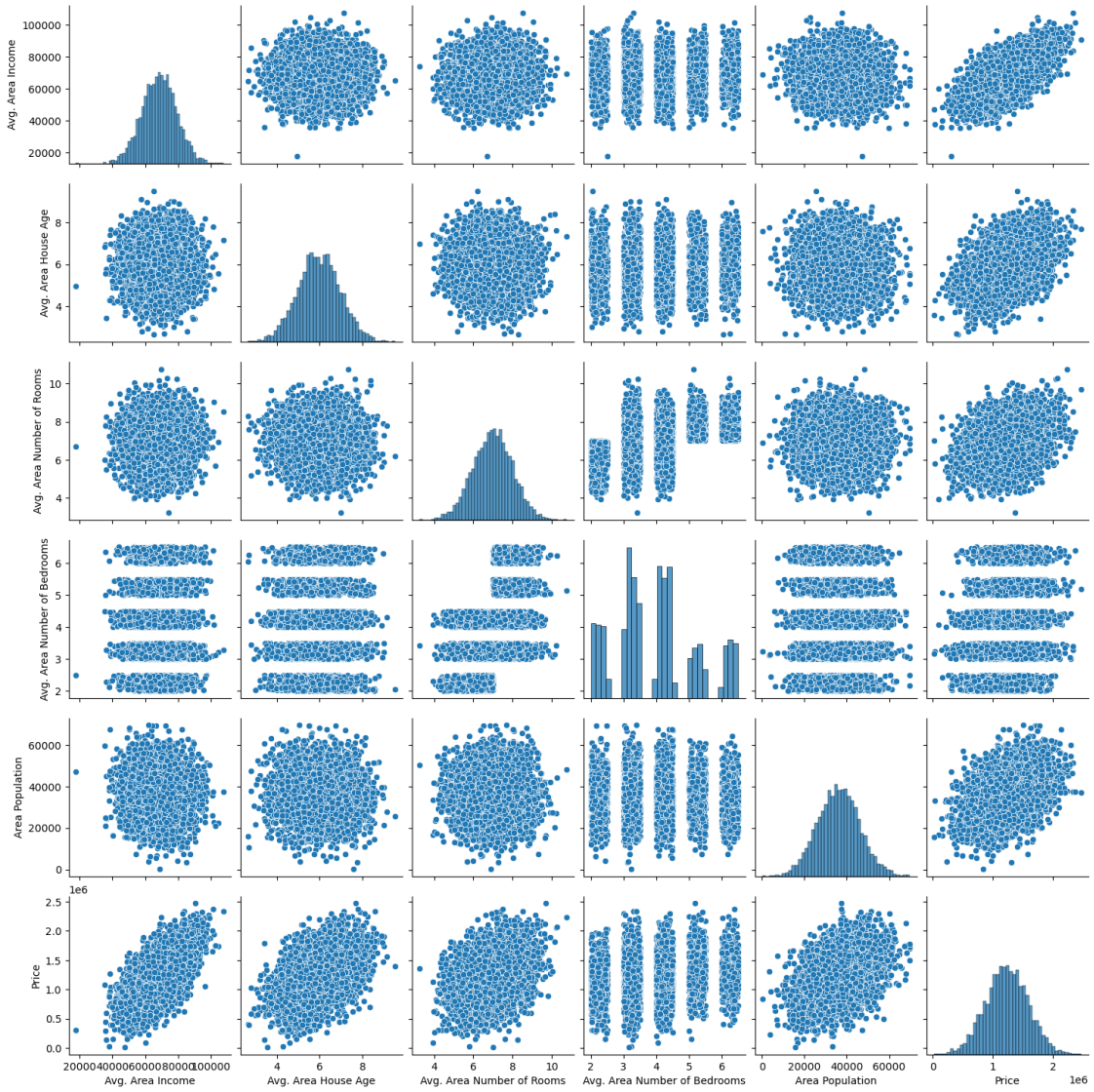
[11]:
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],
      dtype='object')
```

0.1 Exploratory Data Analysis for House Price Prediction

[13]:
```
sns.pairplot(df)
```

```
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

[13]:
```
<seaborn.axisgrid.PairGrid at 0x1ebe20cddd0>
```

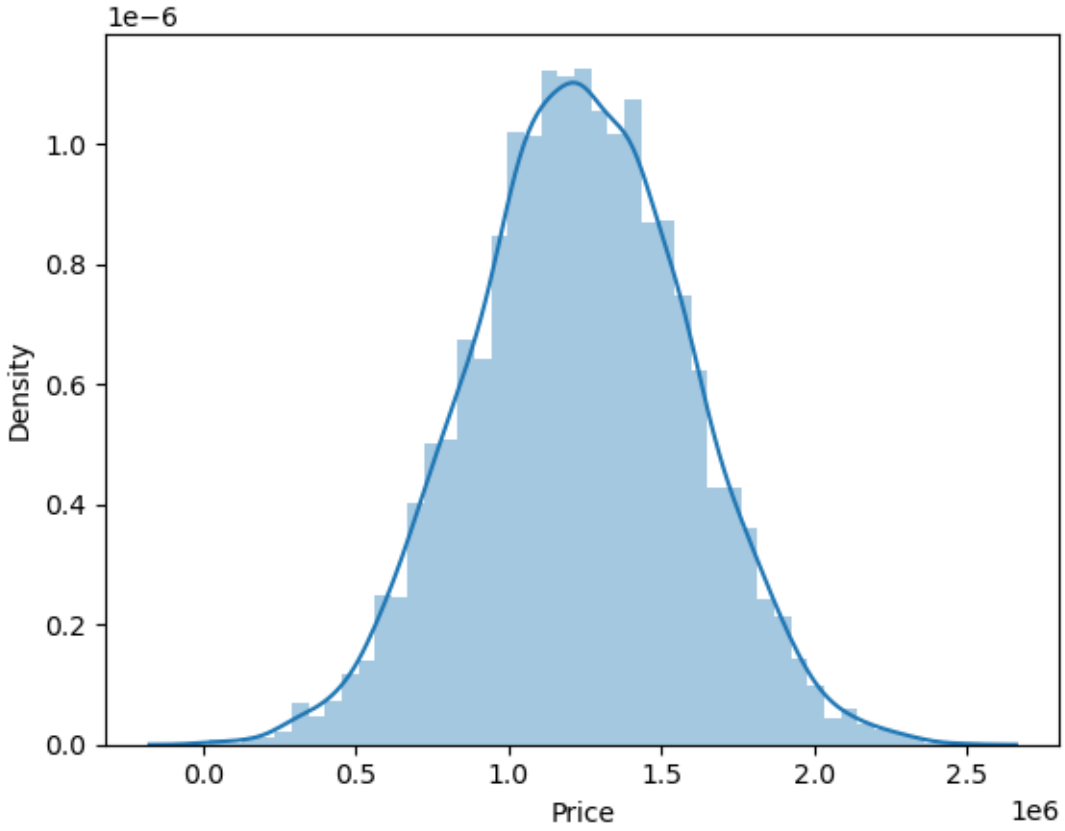


[17]:
```
sns.distplot(df['Price'])
```

```
C:\Users\Anuj\AppData\Local\Temp\ipykernel_20692\834922981.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

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

```
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

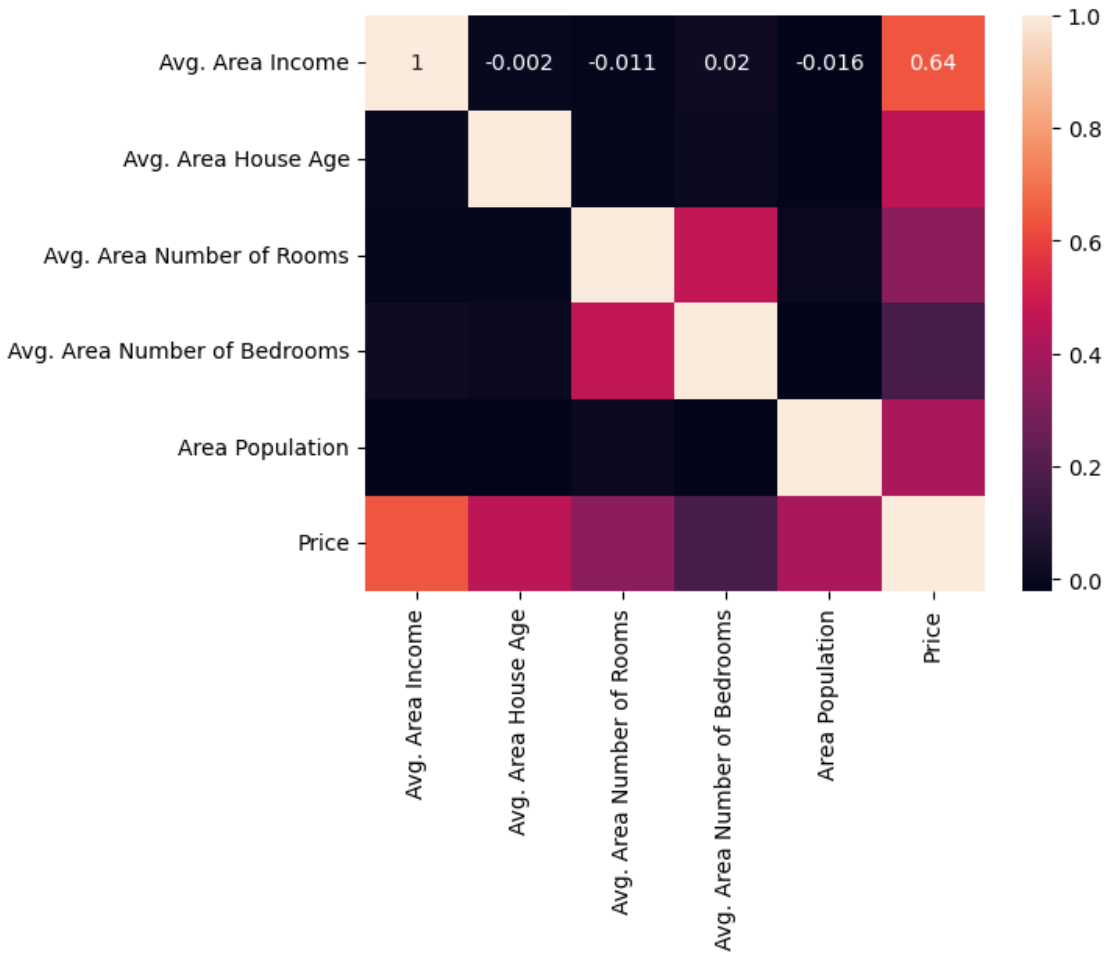
[17]:
```
<Axes: xlabel='Price', ylabel='Density'>
```



[21]:
```
df_numeric = df.drop(columns=['Address'])
```

[23]:
```
sns.heatmap(df_numeric.corr(), annot=True)
```

[23]:
```
<Axes: >
```



0.2 Training a Linear Regression Model

0.2.1 X and y List

[24]:
```
X = df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
        'Avg. Area Number of Bedrooms', 'Area Population']]

y = df['Price']
```

0.2.2 Split Data into Train, Test

[25]:
```
from sklearn.model_selection import train_test_split
```

[26]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,␣
↪random_state=101)
```

0.3 Creating and Training the LinearRegression Model

[27]:
```
from sklearn.linear_model import LinearRegression
```

[28]:
```
lm = LinearRegression()
```

[31]:
```
lm.fit(X_train,y_train)
```

[31]:
```
LinearRegression()
```

0.4 LinearRegression Model Evaluation

[30]:
```
print(lm.intercept_)
```

```
-2640159.796851625
```

[32]:
```
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
coeff_df
```

[32]:

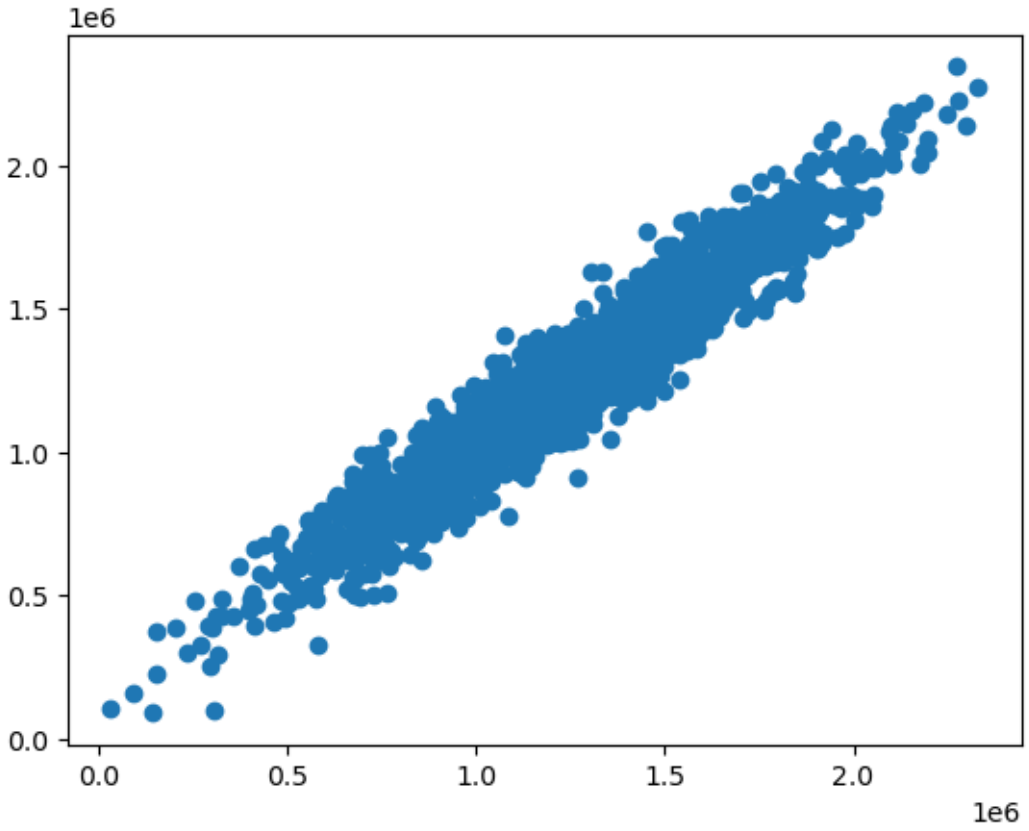
	Coefficient
Avg. Area Income	21.528276
Avg. Area House Age	164883.282027
Avg. Area Number of Rooms	122368.678027
Avg. Area Number of Bedrooms	2233.801864
Area Population	15.150420

0.5 Predictions from our Linear Regression Model

[33]:
```
predictions = lm.predict(X_test)
```

[34]:
```
plt.scatter(y_test,predictions)
```

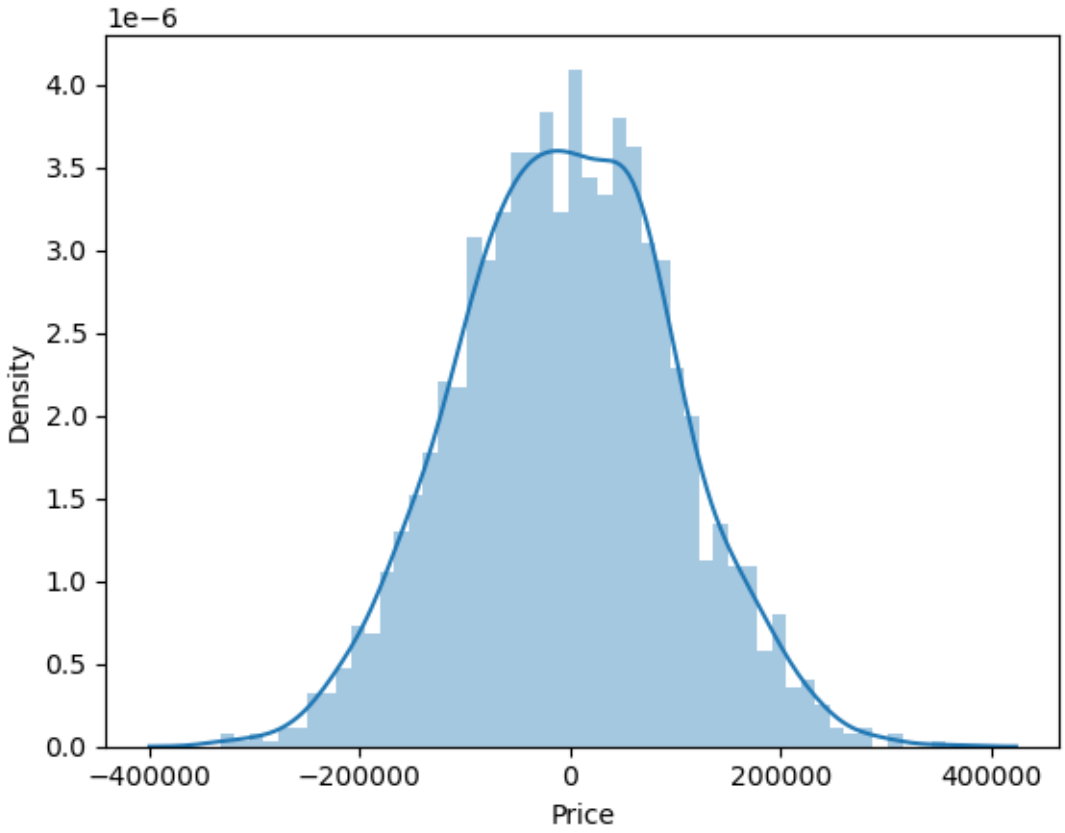
[34]:
```
<matplotlib.collections.PathCollection at 0x1ebec1fce90>
```



0.5.1 In the above scatter plot, we see data is in line shape, which means our model has done good predictions.

[35]:
```
sns.distplot((y_test-predictions),bins=50);
```

```
C:\Users\Anuj\AppData\Local\Temp\ipykernel_20692\1326397652.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((y_test-predictions),bins=50);
C:\Users\Anuj\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



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

0.6 Regression Evaluation Metrics

[36]:
```
from sklearn import metrics
```

[37]:
```
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: 82288.2225191496
MSE: 10460958907.209692
RMSE: 102278.82922291246
```

0.7 THANK YOU! :-)

0.8 Github Link: https://github.com/anujtiwari21?tab=repositories