**Predicting House Prices**

**using Machine Learning**

**Development Part 2**

**INTRODUCTION:**

**House price prediction:**

House price prediction can help the developer determine the selling price of a house and can help the customer to arrange the right time to purchase a house. There are three factors that influence the price of a house which include physical conditions, concept and location.

**Predict house price in machine learning:**

The machine learning model is given the test data but without the price of the properties in order to predict the price for them given the various features for the properties. The predicted price is then compared to the actual price in the test data.

Developing a house price prediction project involves several steps. Here is an outline of the development process:

1. **Data Collection** **:** Collect a large dataset of historical house prices along with relevant features such as location, size, number of bedrooms, etc. This data can be obtained from real estate websites, public records, or through APIs.

**2. Data Preprocessing:** Clean the collected data by removing duplicates, handling missing values, and normalizing numeric features. Also, encode categorical variables using techniques like one-hot encoding or label encoding.

**3. Exploratory Data Analysis (EDA):** Perform EDA to gain insights into the data. Visualize the distribution of house prices and explore correlations between different features. This step helps in understanding the data and identifying any outliers or anomalies.

**4. Feature Engineering:** Create additional features that might be relevant for predicting house prices. For example, you could calculate the price per square foot or create a feature indicating the age of the house based on the year it was built.

**5. Model Selection:** Choose an appropriate machine learning algorithm for house price prediction. Popular choices include linear regression, decision trees, random forests, and gradient boosting algorithms like XGBoost or LightGBM.

**6. Model Training:** Split the dataset into training and testing sets. Train the chosen model on the training set using cross-validation techniques to find the optimal hyperparameters. Evaluate the model's performance using metrics like mean squared error (MSE) or root mean squared error (RMSE).

**7. Model Evaluation:** Test the trained model on the testing set to assess its performance. Compare the predicted house prices with the actual prices and calculate evaluation metrics. Adjust the model if necessary.

**8. Deployment:** Once satisfied with the model's performance, deploy it into a production environment. This could involve creating a web application or an API where users can input house features and get predicted prices as output.

**9. Continuous Improvement:** Monitor the model's performance over time and update it periodically as new data becomes available. This ensures that the predictions remain accurate and up-to-date.

Remember to document each step of the development process and maintain a clean and organized codebase for future reference and collaboration.

**INPUT:**

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import matplotlib.pyplot as plt

import seaborn as sns

!pip install hvplot

import hvplot.pandas

%matplotlib inline

import os

for dirname, \_, filenames **in** os.walk('/kaggle/input'):

for filename **in** filenames:

print(os.path.join(dirname, filename))

USA\_housing = pd.read\_csv("/kaggle/input/usa-housing/USA\_Housing.csv")

USA\_housing.head(10)

USA\_housing.shape

USA\_housing.info()

USA\_housing.describe()

sns.pairplot(USA\_housing, height = 1.5)

USA\_housing.hvplot.hist(by='Price', subplots=False, width=1000)

USA\_housing.hvplot.hist("Price")

USA\_housing.hvplot.scatter(x='Avg. Area House Age', y="Price")

USA\_housing.hvplot.scatter(x='Avg. Area Income',y='Price')

USA\_housing.columns

sns.heatmap(USA\_housing.corr(), annot=True)

X = USA\_housing[['Avg. Area Income','Avg. Area House Age', 'Avg. Area Number of Rooms','Avg. Area Number of Bedrooms','Area Population']]

y = USA\_housing['Price']

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, random\_state=42)

from sklearn import metrics

from sklearn.model\_selection import cross\_val\_score

def cross\_val(model):

pred = cross\_val\_score(model, X, y , cv=10)

return pred.mean()

def print\_evaluate(true, predicted):

mae= metrics.mean\_absolute\_error(true, predicted)

mse= metrics.mean\_squared\_error(true, predicted)

rmse=np.sqrt(metrics.mean\_squared\_error(true, predicted))

r2\_square = metrics.r2\_score(true, predicted)

print("MAE",mae)

print("MSE", mse)

print("RMSE",rmse)

print("R2 Square",r2\_square)

print('----------------------------')

def evaluate(true, predicted):

mae= metrics.mean\_absolute\_error(true, predicted)

mse= metrics.mean\_squared\_error(true, predicted)

rmse = np.sqrt(metrics.mean\_squared\_error(true,predicted))

r2\_square = metrics.r2\_score(true, predicted)

return mae, mse, rmse, r2\_square

from sklearn.preprocessing import StandardScaler

from sklearn.pipeline import Pipeline

pipeline = Pipeline([

('std\_scalar', StandardScaler())

])

X\_train = pipeline.fit\_transform(X\_train)

X\_test = pipeline.transform(X\_test)

from sklearn.linear\_model import LinearRegression

lin\_reg = LinearRegression()

lin\_reg.fit(X\_train, y\_train)

print(lin\_reg.intercept\_)

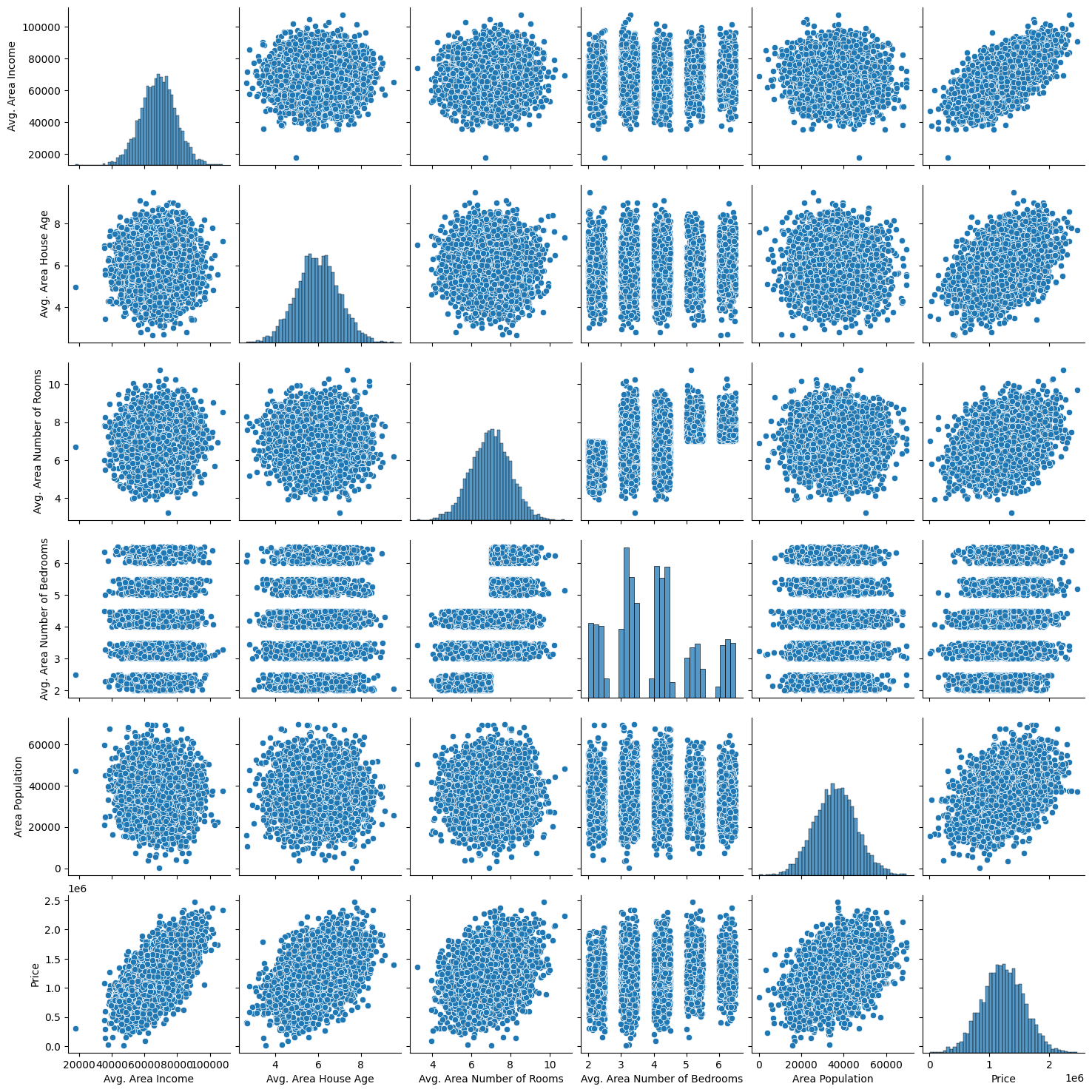
coeff\_df = pd.DataFrame(lin\_reg.coef\_, X.columns, columns=['Coefficient'])

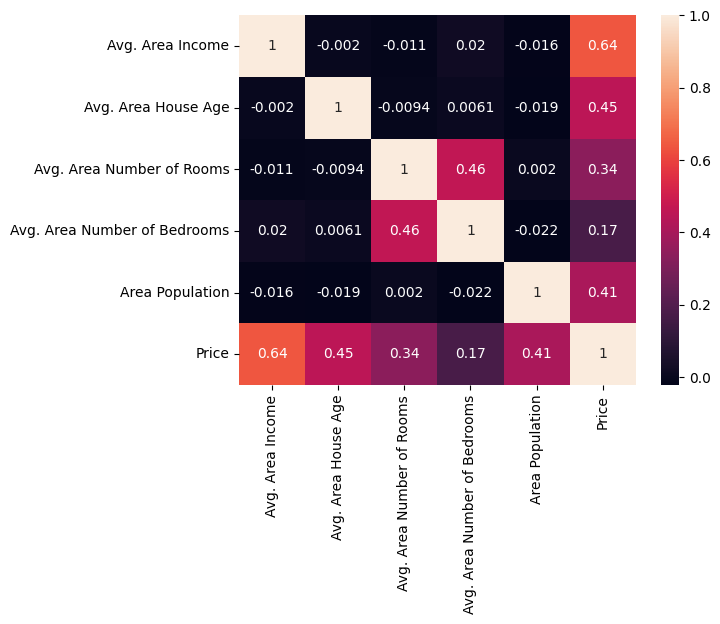
coeff\_df

**OUTPUT :**

|  | Avg. Area Income | Avg. Area House Age | Avg. Area Number of Bedrooms | 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 |

|  | count | mean | std | min | 25% | 50% | 75% | max |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Avg. Area Income | 5000.0 | 6.858311e+04 | 10657.991214 | 17796.631190 | 61480.562388 | 6.880429e+04 | 7.578334e+04 | 1.077017e+05 |
| Avg. Area House Age | 5000.0 | 5.977222e+00 | 0.991456 | 2.644304 | 5.322283 | 5.970429e+00 | 6.650808e+00 | 9.519088e+00 |
| Avg. Area Number of Rooms | 5000.0 | 6.987792e+00 | 1.005833 | 3.236194 | 6.299250 | 7.002902e+00 | 7.665871e+00 | 1.075959e+01 |
| Avg. Area Number of Bedrooms | 5000.0 | 3.981330e+00 | 1.234137 | 2.000000 | 3.140000 | 4.050000e+00 | 4.490000e+00 | 6.500000e+00 |
| Area Population | 5000.0 | 3.616352e+04 | 9925.650114 | 172.610686 | 29403.928702 | 3.619941e+04 | 4.286129e+04 | 6.962171e+04 |
| Price | 5000.0 | 1.232073e+06 | 353117.626581 | 15938.657923 | 997577.135049 | 1.232669e+06 | 1.471210e+06 | 2.469066e+06 |





|  | Variables | vif |
| --- | --- | --- |
| 0 | Avg. Area Income | 1.001159 |
| 1 | Avg. Area House Age | 1.000577 |
| 2 | Avg. Area Number of Rooms | 1.273535 |
| 3 | Avg. Area Number of Bedrooms | 1.274413 |
| 4 | Area Population | 1.001266 |

|  | Model | RMSE | R2 | Verification |
| --- | --- | --- | --- | --- |
| 0 | Linear model | 80879.0972 | 0.9180 | 0.9174 |
| 1 | Ridge model | 80878.9638 | 0.9180 | 0.9174 |
| 2 | Lasso model | 80879.0910 | 0.9180 | 0.9174 |
| 3 | ElasticNet model | 81617.9048 | 0.9157 | 0.9165 |

**Conclusion :**

In conclusion, developing a house price prediction project involves collecting and preprocessing data, performing exploratory data analysis, engineering relevant features, selecting and training a suitable machine learning model, evaluating its performance, deploying the model, and continuously improving it over time. By following these steps and maintaining good documentation and code organization, you can create an accurate and reliable house price prediction system.