House Price Prediction

INTRODUCTION:

What is 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.

What is the prediction of housing prices in India?

Property prices in India are expected to increase 7.5% on a pan-India basis this year, the fastest growth in five years, according to a Reuters poll of property analysts. Average house prices were forecast to rise 6% next year and in 2024. The poll of 13 property analysts were held during May11-27.

How to 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.

IMPORTANT DEPENDENCIES:

Exploratory Data Analysis:

EDA refers to the deep analysis of data so as to discover different patterns and spot anomalies. Before making inferences from data, it is essential to examine all your variables.
So here let's make a heatmap using seaborn library.

Data Cleaning:

Data Cleaning is the way to improvise the data or remove incorrect, corrupted or irrelevant data.

As in our dataset, there are some columns that are not important and irrelevant for the model training. So, we can drop that column before training. There are 2 approaches to dealing with empty/null values

- We can easily delete the column/row (if the feature or record is not much important).
- Filling the empty slots with mean/mode/0/NA/etc. (depending on the dataset requirement).

Model and Accuracy:

As we have to train the model to determine the continuous values, so we will be using these regression models.

- SVM-Support Vector Machine
- Random Forest Regressor
- Linear Regressor

Competition Description:

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

Practice Skills:

- Creative feature engineering
- Advanced regression techniques like random forest and gradient boosting

Goal:

It is your job to predict the sales price for each house. For each Id in the test set, you must predict the value of the Sale Price variable.

INPUT:

```
import pandas as pd import
numpy as np import seaborn as
sns import matplotlib.pyplot as
plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_absolute_error,mean_squared_error
from sklearn.linear_model import LinearRegression from sklearn.linear_model
import Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR import xgboost as xg

%matplotlib inline import
warnings
warnings.filterwarnings("ignore")
```

```
dataset = pd.read_csv('/kaggle/input/usa-housing/USA_Housing.csv')
dataset   dataset.info()   dataset.describe()   sns.histplot(dataset,
x='Price',   bins=50,   color='y')   sns.boxplot(dataset,   x='Price',
palette='Blues')  sns.jointplot(dataset,  x='Avg.  Area  House  Age',
y='Price',  kind='hex')  sns.jointplot(dataset,  x='Avg.  Area  Income',
y='Price') plt.figure(figsize=(12,8))
```

```
sns.pairplot(dataset)
dataset.hist(figsize=(10,8))
dataset.corr(numeric_only=True)
plt.figure(figsize=(10,5))
sns.heatmap(dataset.corr(numeric_only = True), annot=True)
X = dataset[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
      'Avg. Area Number of Bedrooms', 'Area Population']]
Y_train.head()
Y = dataset['Price']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=101)
Y_train.head()
Y_train.shape
Y_test.head()
Y_test.shape


sc = StandardScaler()
X_train_scal = sc.fit_transform(X_train)
X_test_scal = sc.fit_transform(X_test)
model_lr=LinearRegression()
model_lr.fit(X_train_scal, Y_train)
Prediction1 = model_lr.predict(X_test_scal)
   plt.figure(figsize=(12,6))
   plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')
plt.plot(np.arange(len(Y_test)), Prediction1, label='Predicted Trend')
plt.xlabel('Data')     plt.ylabel('Trend')     plt.legend()
   plt.title('Actual vs Predicted')  sns.histplot((Y_test-
Prediction1), bins=50) print(r2_score(Y_test,
Prediction1))  print(mean_absolute_error(Y_test,
Prediction1)) print(mean_squared_error(Y_test,
Prediction1)) model_svr = SVR()
model_svr.fit(X_train_scal, Y_train)
Prediction2 = model_svr.predict(X_test_scal)
   plt.figure(figsize=(12,6))
   plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')
plt.plot(np.arange(len(Y_test)), Prediction2, label='Predicted Trend')
plt.xlabel('Data')     plt.ylabel('Trend')     plt.legend()
   plt.title('Actual vs Predicted')  sns.histplot((Y_test-
Prediction2), bins=50) print(r2_score(Y_test,
Prediction2))  print(mean_absolute_error(Y_test,
Prediction2)) print(mean_squared_error(Y_test,
Prediction2)) model_lar = Lasso(alpha=1)
model_lar.fit(X_train_scal,Y_train)
Prediction3 = model_lar.predict(X_test_scal)
   plt.figure(figsize=(12,6))
   plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')



   plt.plot(np.arange(len(Y_test)), Prediction3, label='Predicted Trend')
plt.xlabel('Data')     plt.ylabel('Trend')     plt.legend()
   plt.title('Actual vs Predicted') sns.histplot((Y_test-
Prediction3), bins=50) print(r2_score(Y_test,
```

```
Prediction2)) print(mean_absolute_error(Y_test,
Prediction2)) print(mean_squared_error(Y_test,
Prediction2)) model_rf =
RandomForestRegressor(n_estimators=50)
model_rf.fit(X_train_scal, Y_train)
Prediction4 = model_rf.predict(X_test_scal)
    plt.figure(figsize=(12,6))
    plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')
plt.plot(np.arange(len(Y_test)), Prediction4, label='Predicted Trend')
plt.xlabel('Data')     plt.ylabel('Trend')     plt.legend()     plt.title('Actual
vs Predicted')

sns.histplot((Y_test-Prediction4), bins=50) print(r2_score(Y_test, Prediction2))
print(mean_absolute_error(Y_test, Prediction2)) print(mean_squared_error(Y_test, Prediction2))


model_xg = xg.XGBRegressor()

model_xg.fit(X_train_scal, Y_train)
```

```
Prediction5 = model_xg.predict(X_test_scal)
    plt.figure(figsize=(12,6))
    plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')
plt.plot(np.arange(len(Y_test)), Prediction5, label='Predicted Trend')
plt.xlabel('Data')     plt.ylabel('Trend')     plt.legend()
    plt.title('Actual vs Predicted') sns.histplot((Y_test-
Prediction4), bins=50) print(r2_score(Y_test, Prediction2))
print(mean_absolute_error(Y_test, Prediction2))
print(mean_squared_error(Y_test, Prediction2))
### Linear Regression is giving us best Accuracy.
```

Output:

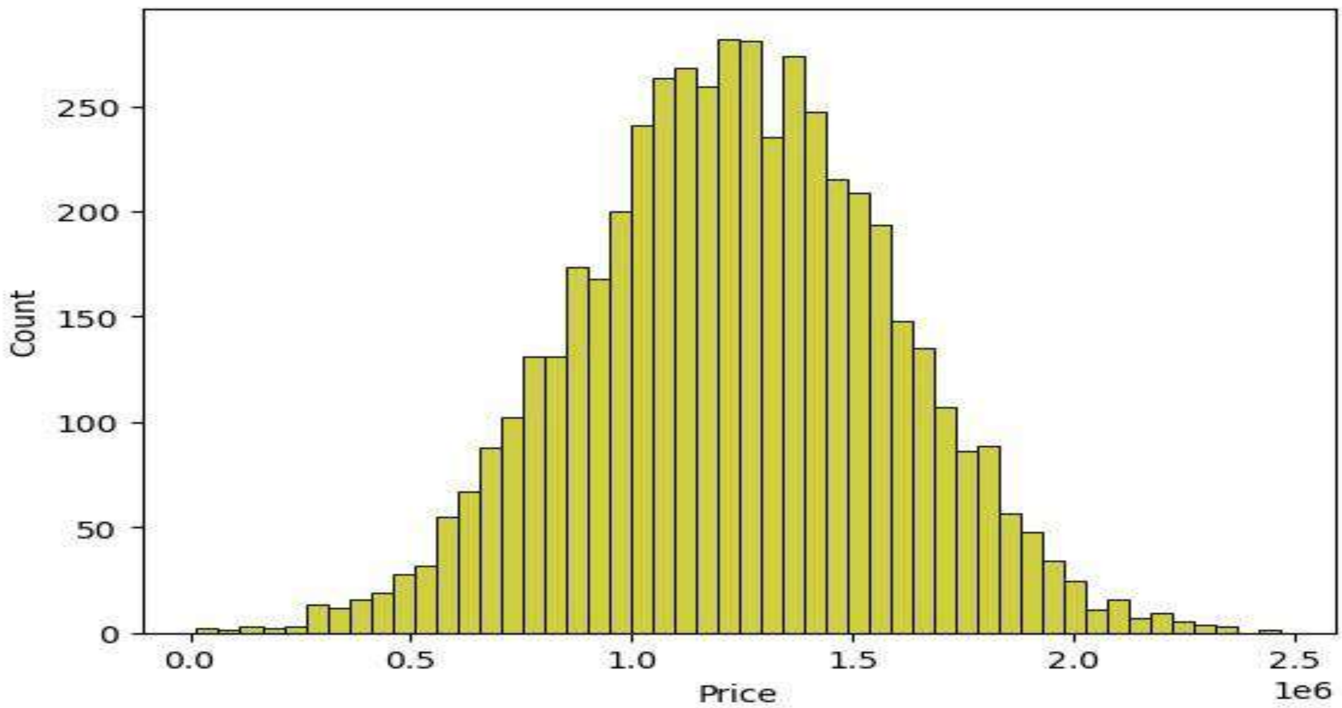
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.80003	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 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
...	...	...	...	...	...	...	...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFPO AP 30153-7653
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42991- 3352
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garden Suite 076\nJoshualand, VA 01...
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO AE 73316
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06 37778	37778 George Ridges Apt. 509\nEast Holly, NV 2...

	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

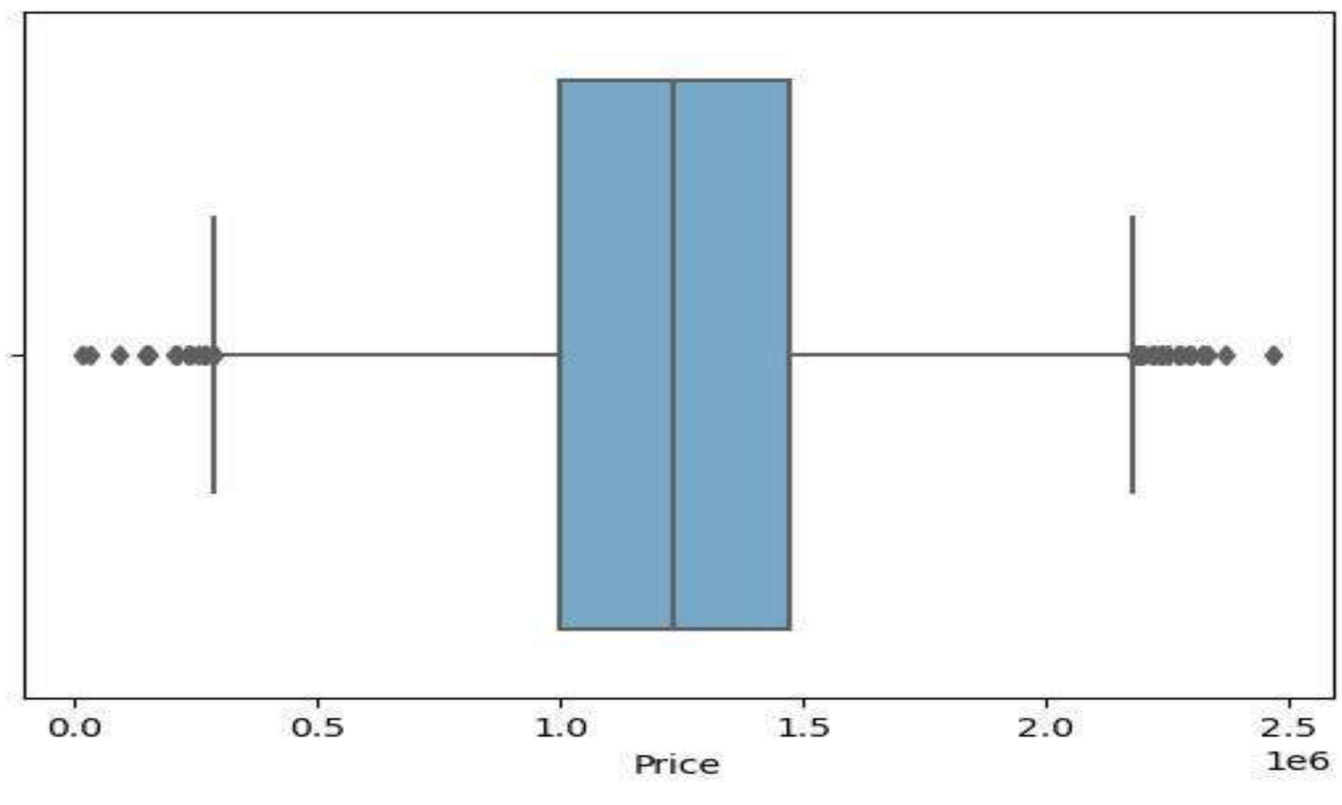
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',

```
    'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],
dtype='object')
```

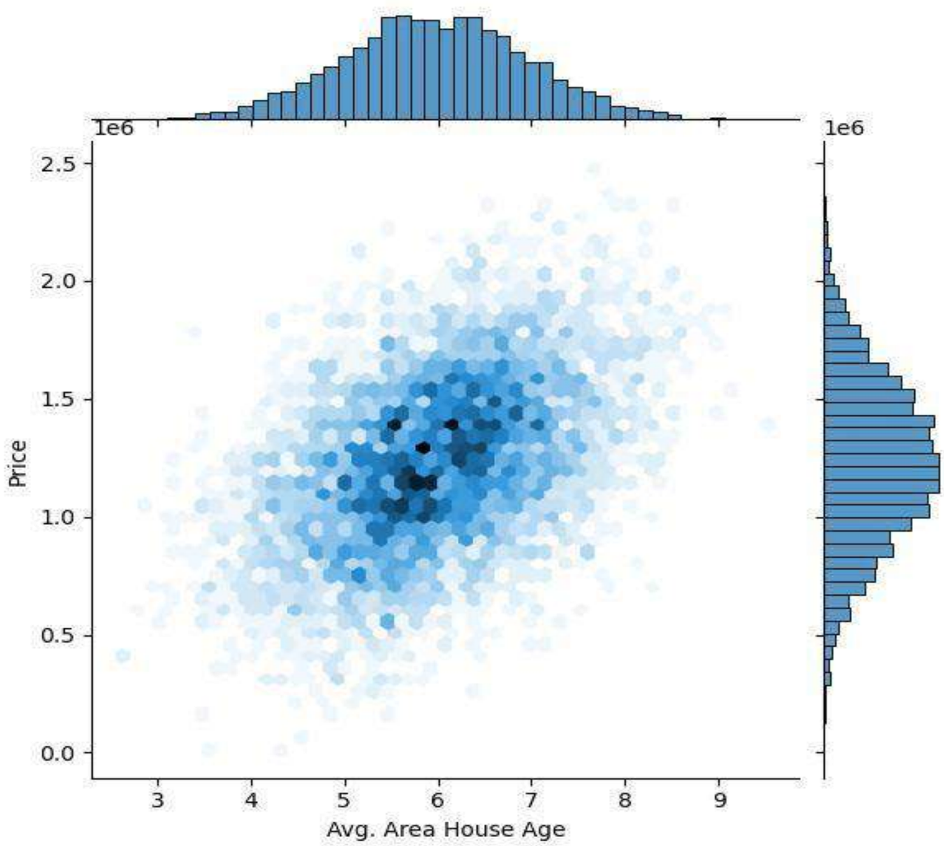
<Axes: xlabel='Price', ylabel='Count'>



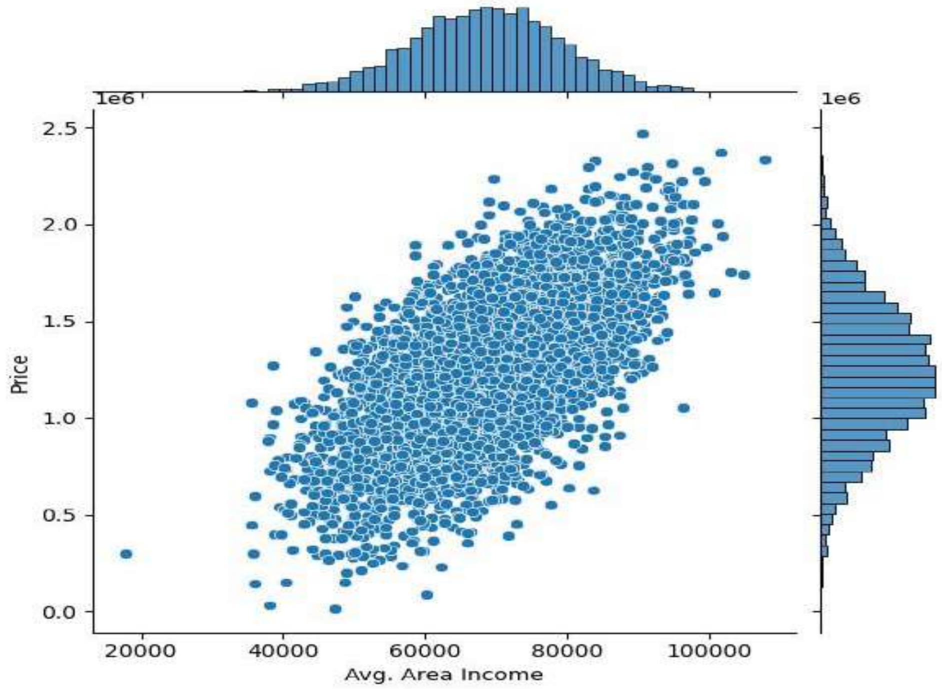
<Axes: xlabel='Price'>



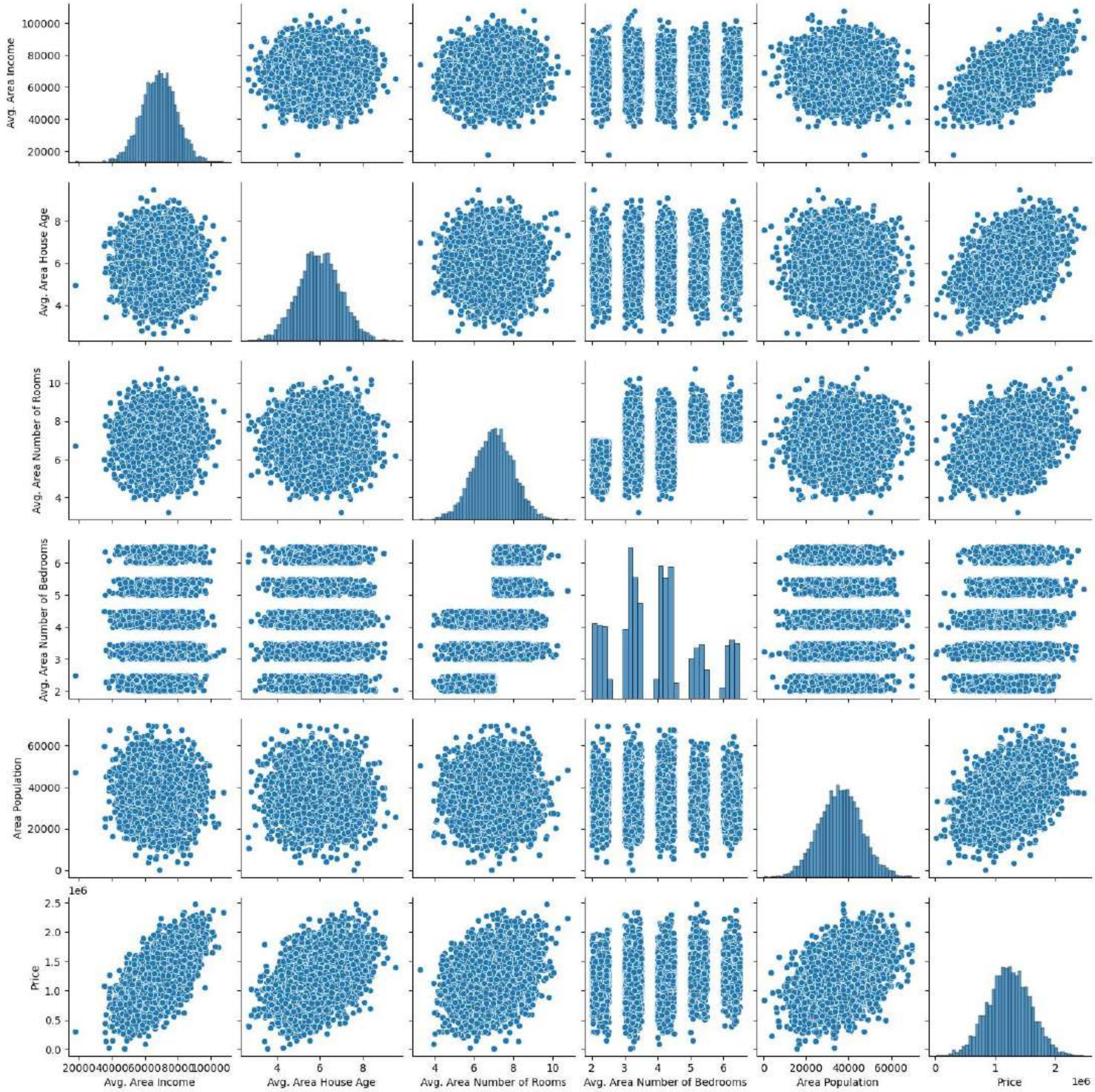
<seaborn.axisgrid.JointGrid at 0x7dbe246100a0>



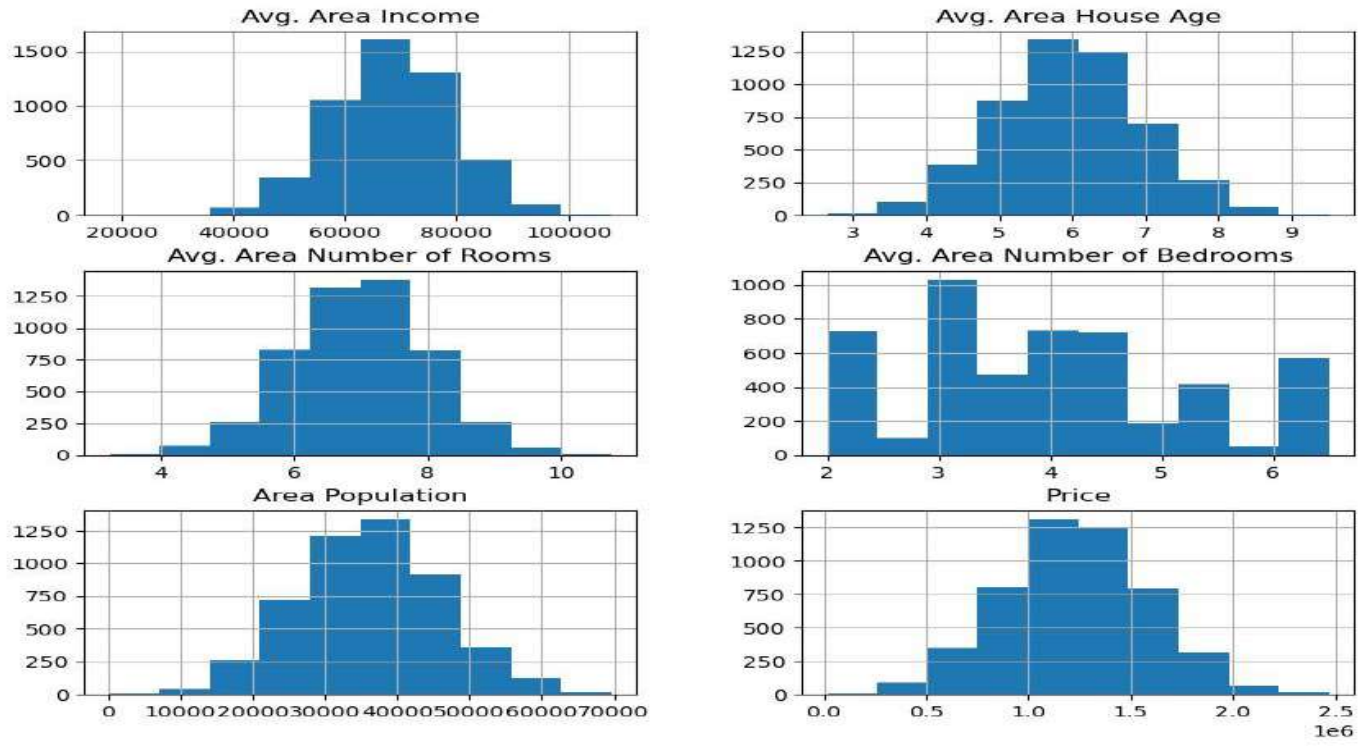
<seaborn.axisgrid.JointGrid at 0x7dbe1333c250>



<seaborn.axisgrid.PairGrid at 0x7dbe1333c340>
<Figure size 1200x800 with 0 Axes>

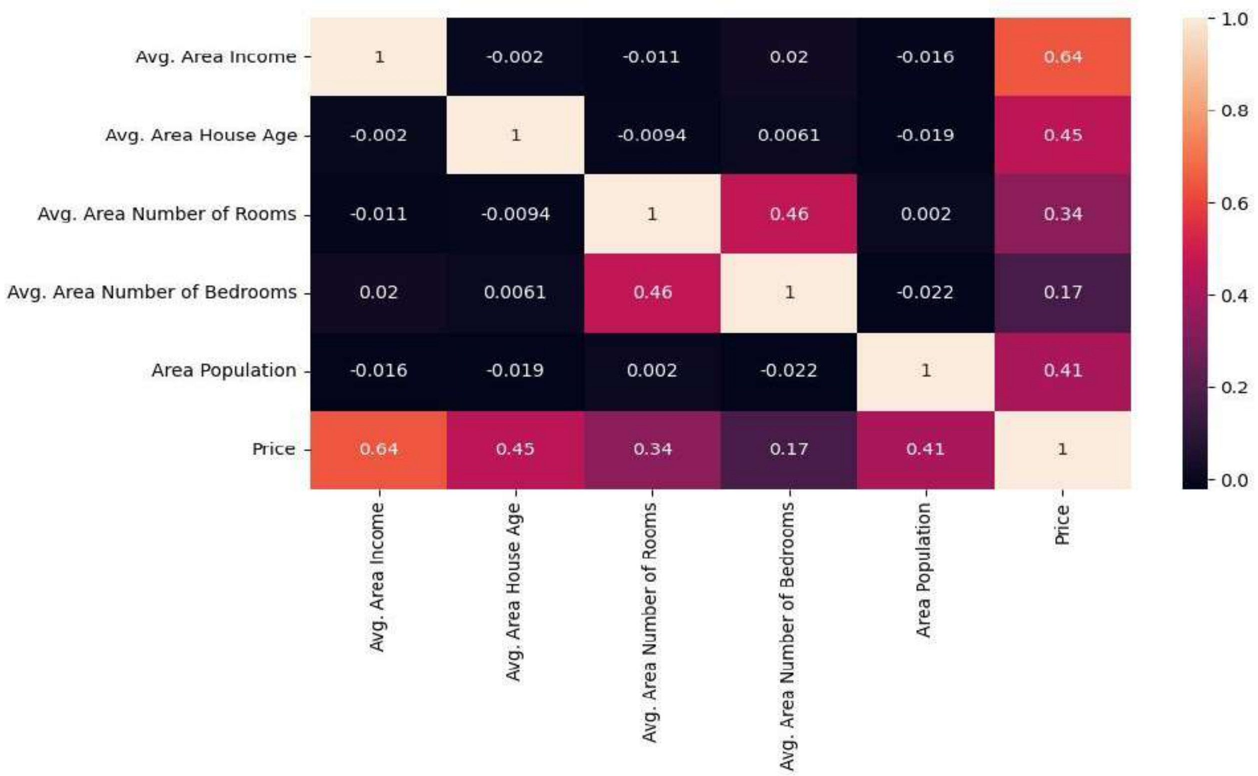



```
array([[<Axes: title={'center': 'Avg. Area Income'}>,
<Axes: title={'center': 'Avg. Area House Age'}>],
      [<Axes: title={'center': 'Avg. Area Number of Rooms'}>,
       <Axes: title={'center': 'Avg. Area Number of Bedrooms'}>],
      [<Axes: title={'center': 'Area Population'}>,
       <Axes: title={'center': 'Price'}>]], dtype=object)
```



	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

<Axes: >

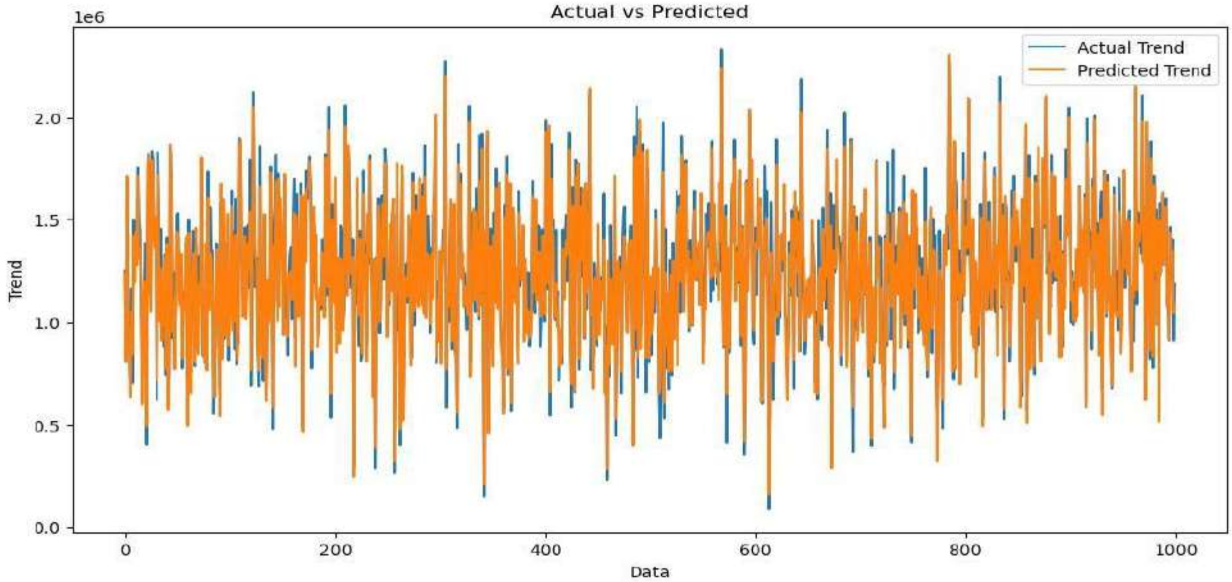



```
3413    1.305210e+06
1610    1.400961e+06
3459    1.048640e+06
4293    1.231157e+06
1039    1.391233e+06
Name: Price, dtype: float64

(4000,)

1718    1.251689e+06
2511    8.730483e+05
345     1.696978e+06
2521    1.063964e+06
54      9.487883e+05
Name: Price, dtype: float64

(1000,)
```

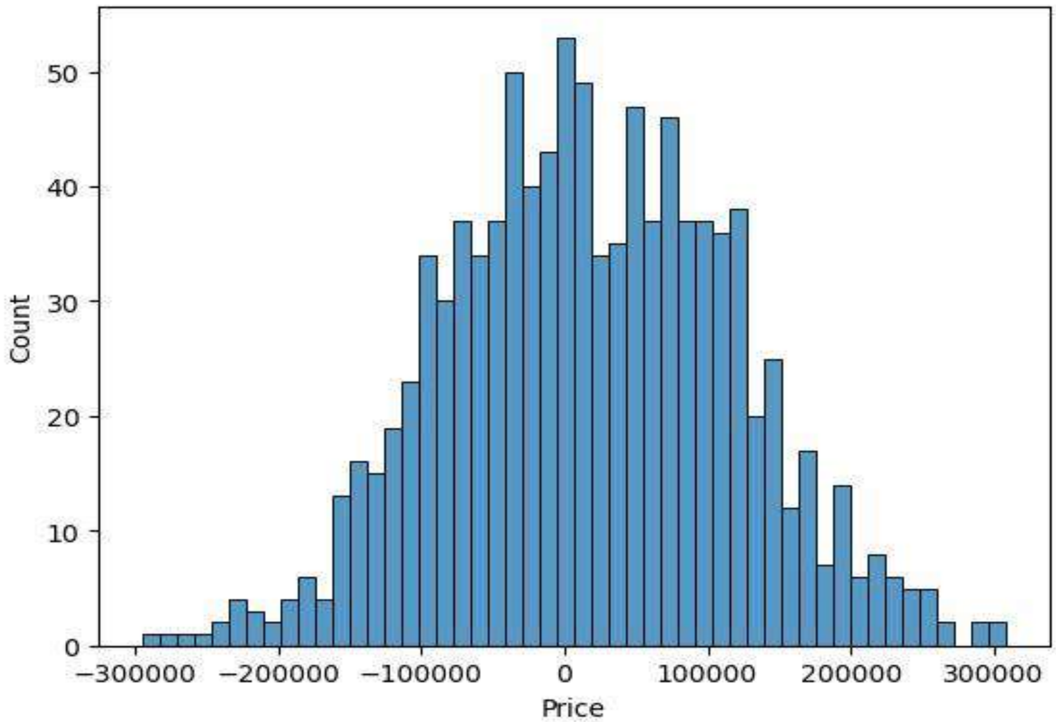
LinearRegression

LinearRegression()

Text(0.5, 1.0, 'Actual vs Predicted')



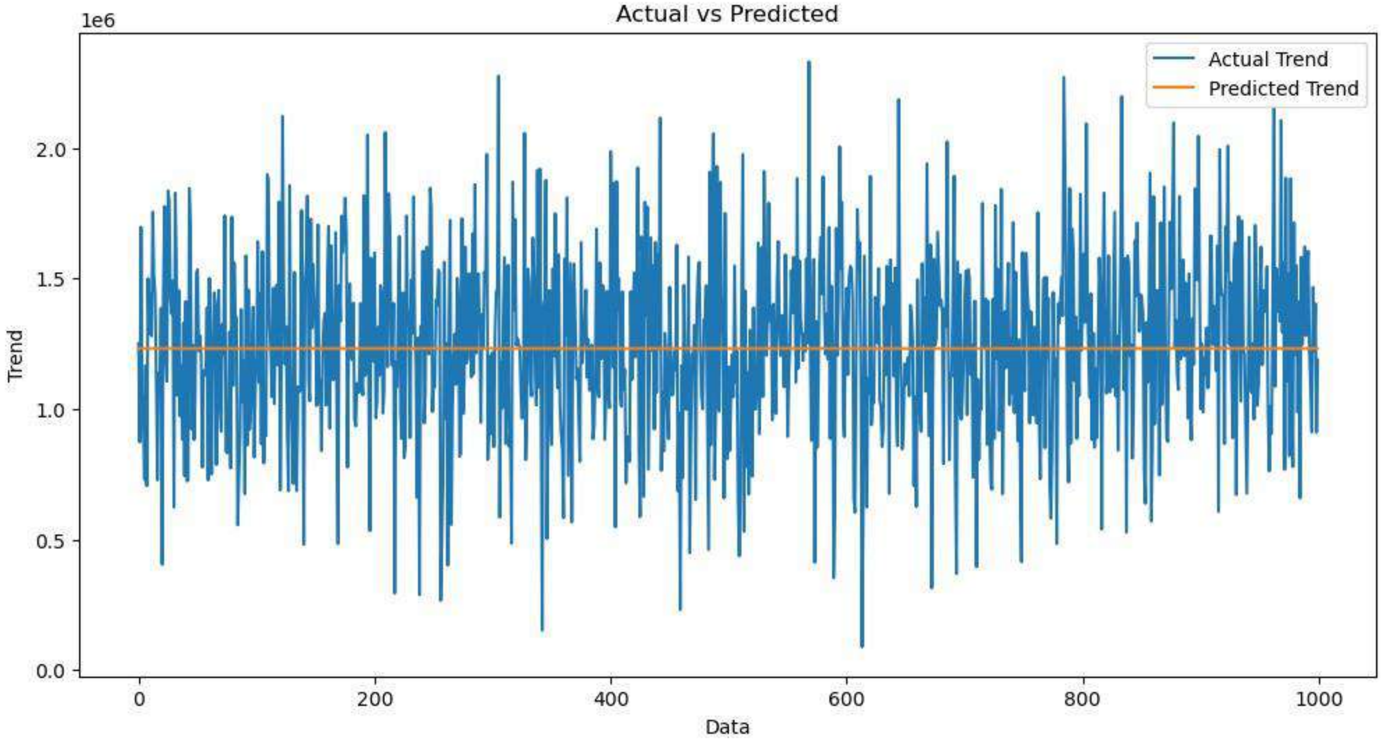
<Axes: xlabel='Price', ylabel='Count'>



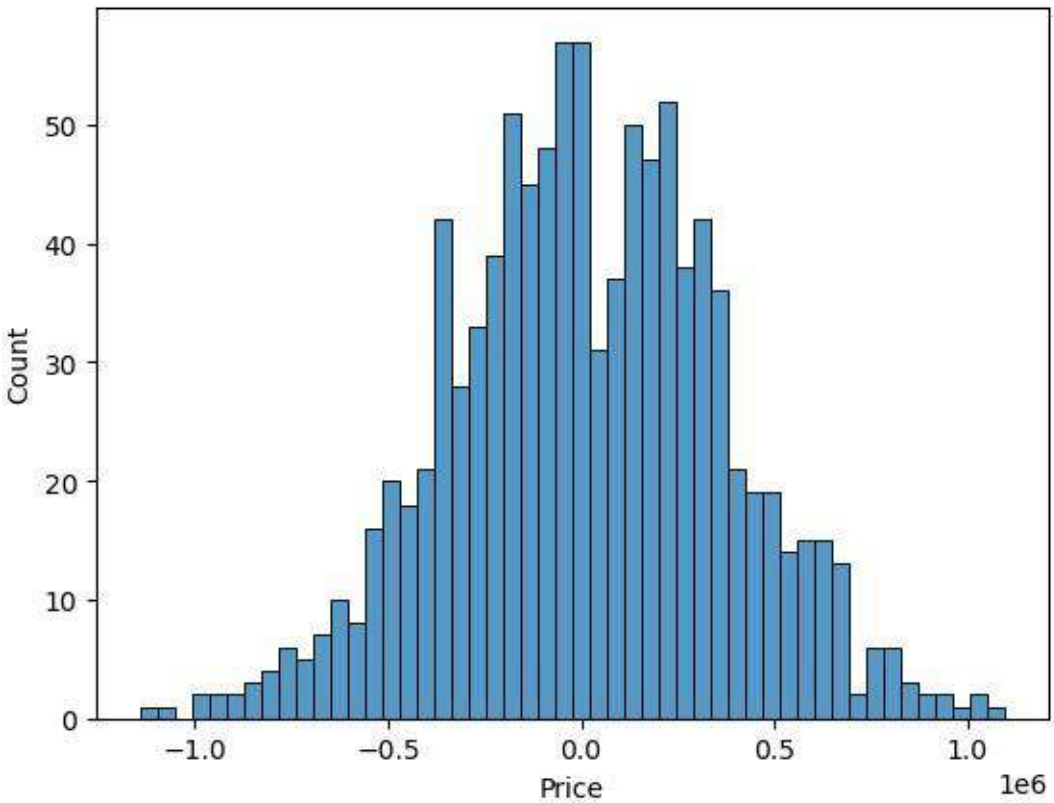
SVR

SVR()

Text(0.5, 1.0, 'Actual vs Predicted')



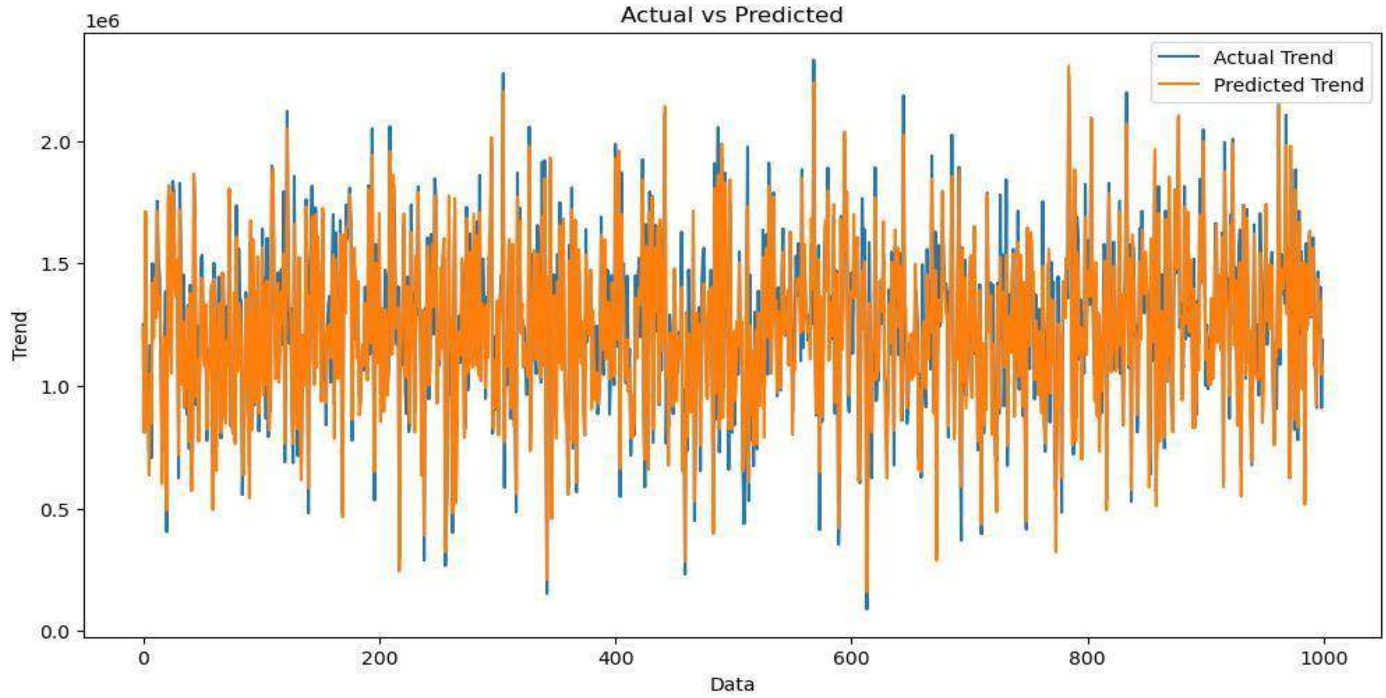
<Axes: xlabel='Price', ylabel='Count'>



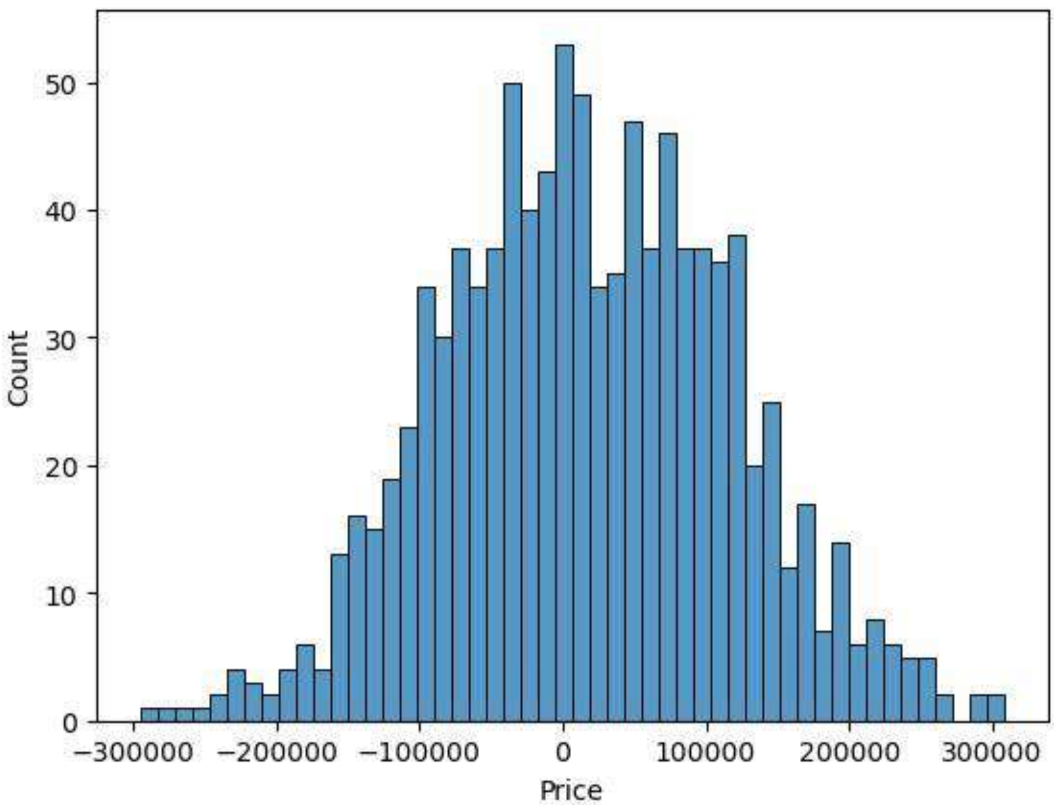
Lasso

Lasso(alpha=1)

Text(0.5, 1.0, 'Actual vs Predicted')



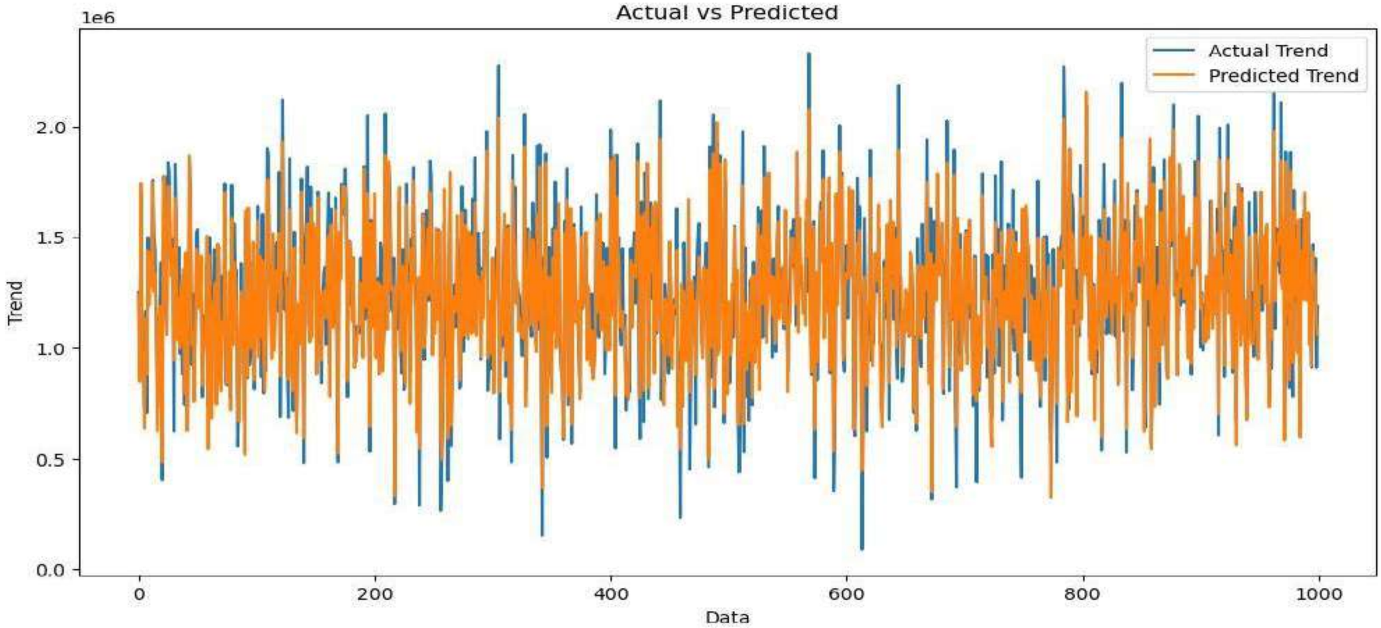
<Axes: xlabel='Price', ylabel='Count'>



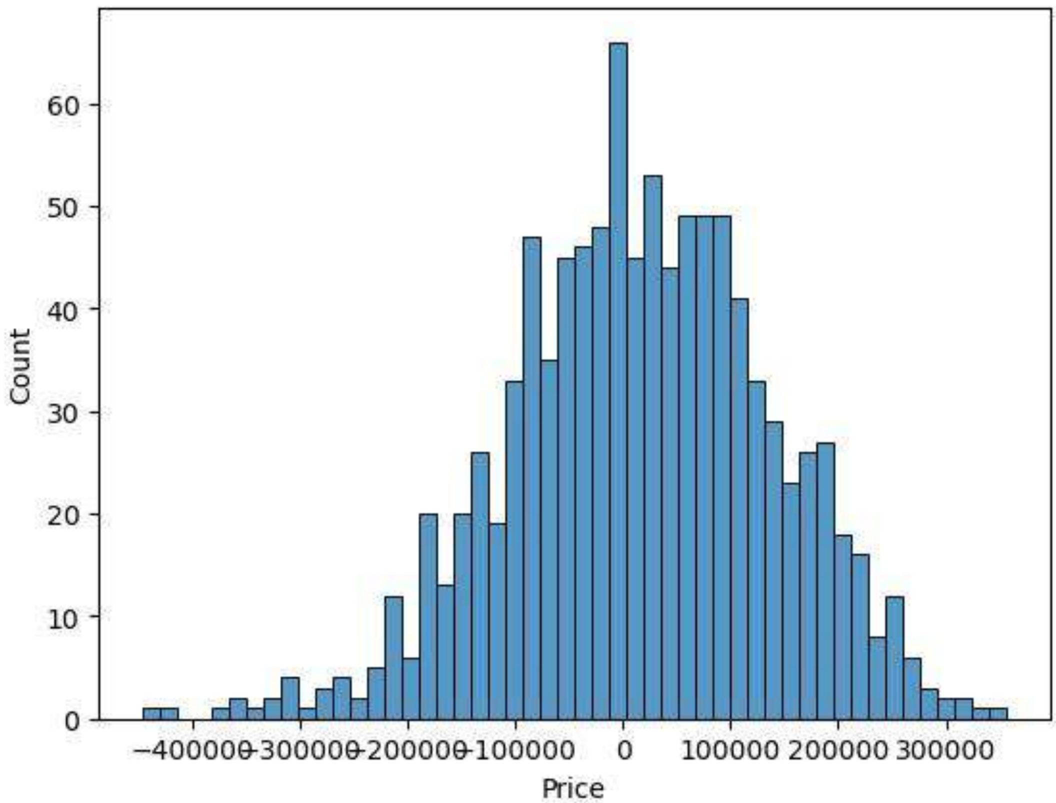
RandomForestRegressor

RandomForestRegressor(n_estimators=50)

Text(0.5, 1.0, 'Actual vs Predicted')



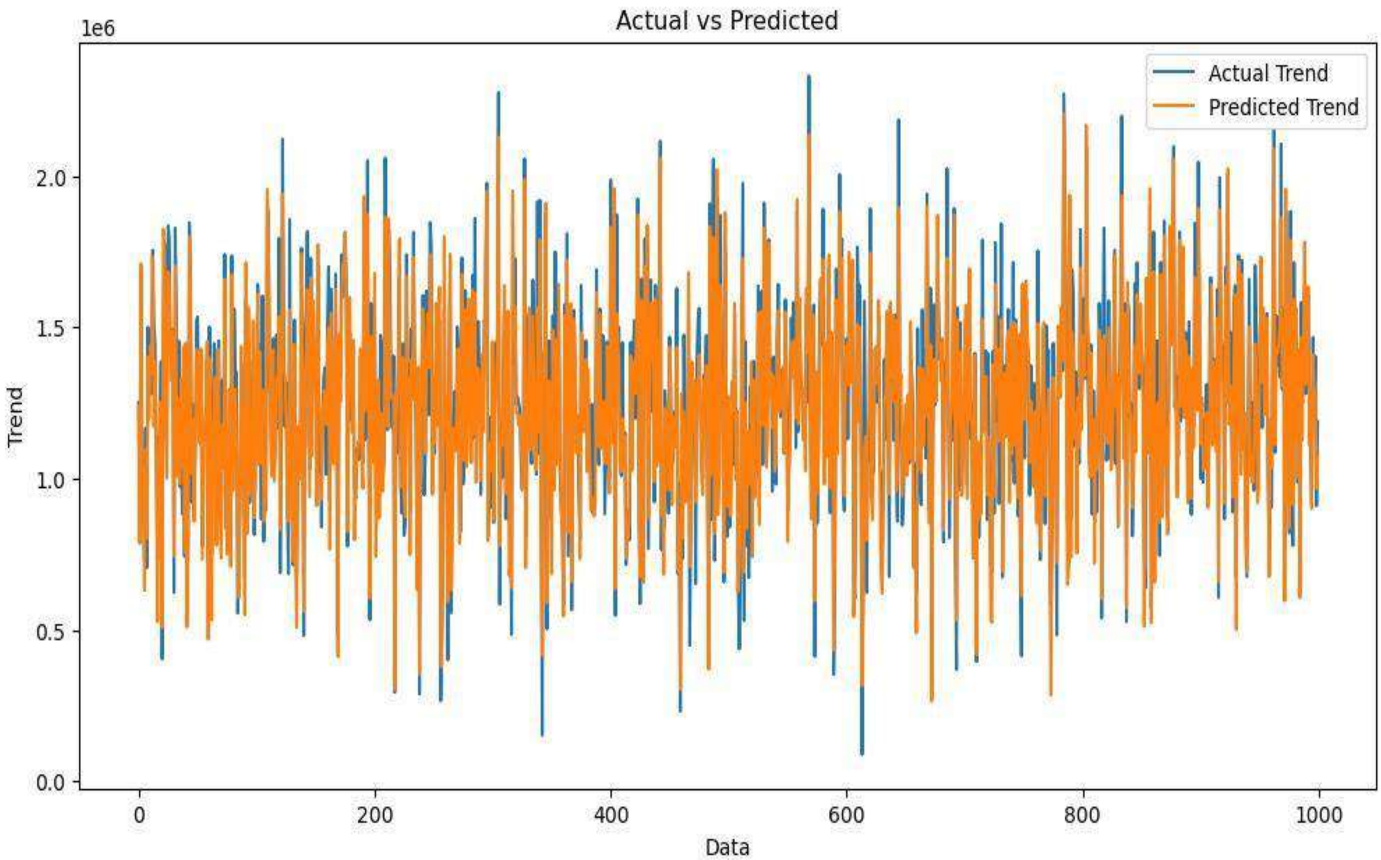
<Axes: xlabel='Price', ylabel='Count'>



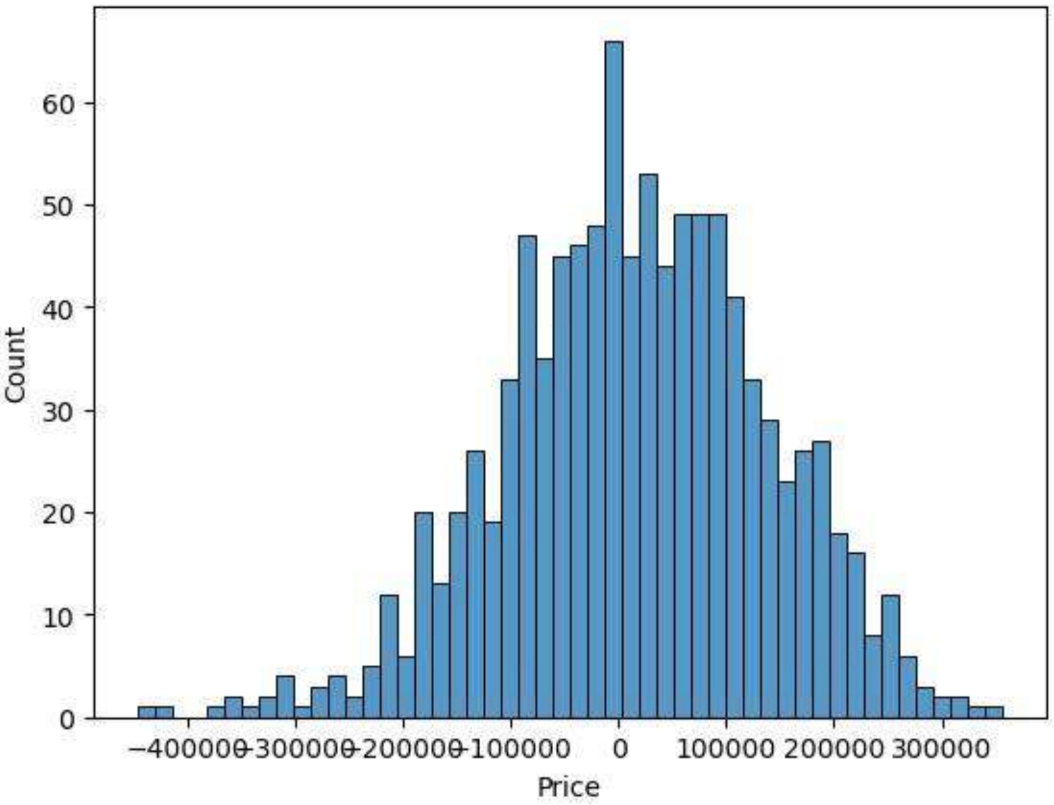
XGBRegression

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bynode=None,              colsample_bytree=None,
early_stopping_rounds=None,              enable_categorical=False, eval_metric=None,
feature_types=None,              gamma=None, gpu_id=None, grow_policy=None,
importance_type=None,              interaction_constraints=None,
learning_rate=None, max_bin=None,              max_cat_threshold=None,
max_cat_to_onehot=None,              max_delta_step=None, max_depth=None,
max_leaves=None,              min_child_weight=None, missing=nan,
monotone_constraints=None,              n_estimators=100, n_jobs=None,
num_parallel_tree=None,              predictor=None, random_state=None, ...)
```

Text(0.5, 1.0, 'Actual vs Predicted')



<Axes: xlabel='Price', ylabel='Count'>



CONCLUSION:

To conclude, the application of machine learning in property research is still at an early stage. We hope this study has moved a small step ahead in providing some methodological and empirical contributions to property appraisal, and presenting an alternative approach to the valuation of housing prices. Future direction of research may consider incorporating additional property transaction data from a larger geographical location with more features, or analyzing other property types beyond housing development.