PREDICTING HOUSE PRICE USING MACHINE LEARNING

Phase 2 Submission document

Project: House price prediction using machine learning

Introduction:

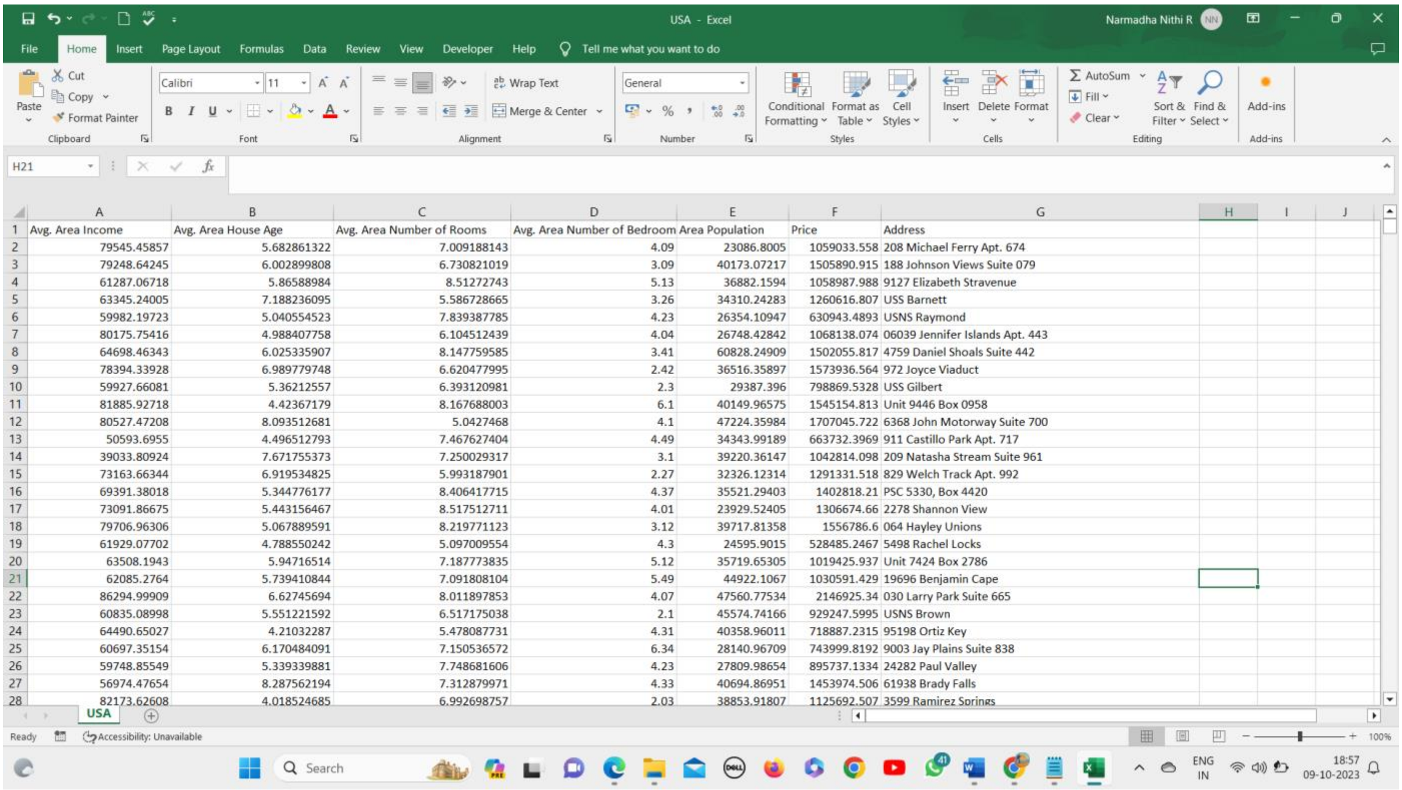
- In recent years, the real estate market has witnessed a significant transformation, driven by advancements in technology and data analytics. One of the most exciting developments in this field is the application of machine learning and advanced techniques to predict house prices accurately. Whether you are a homebuyer looking to make an informed decision or a real estate professional aiming to gain a competitive edge, understanding how machine learning can be employed to forecast house prices is a valuable skill.
- Traditional techniques for predicting house prices include multiple linear regression and decision trees. These methods rely on historical sales data and straightforward mathematical relationships between features and prices.
- Advanced techniques employ machine learning algorithms like Random Forests, Gradient Boosting, and Support Vector Machines. They can model complex relationships and handle large datasets effectively. Additionally, deep learning techniques such as neural networks are gaining popularity for their ability to capture intricate patterns in the data. Time series analysis and ensemble methods further enhance prediction accuracy by considering temporal trends and combining multiple

models. These advanced techniques offer more sophisticated and accurate house price predictions, benefiting both buyers and sellers in the real estate market.

Data source:

A good data source for house price prediction using machine learning should be accurate, complete, covering the geographic area of interest and must be accessible.

Dataset link: (https://www.kaggle.com/datasets/vedavyasv/usa-housing)



	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedroom	Area Population	Price	Address
2	79545.45857	5.682861322	7.009188143	4.09	23086.8005	1059033.558	208 Michael Ferry Apt. 674
3	79248.64245	6.002899808	6.730821019	3.09	40173.07217	1505890.915	188 Johnson Views Suite 079
4	61287.06718	5.86588984	8.51272743	5.13	36882.1594	1058987.988	9127 Elizabeth Stravenue
5	63345.24005	7.188236095	5.586728665	3.26	34310.24283	1260616.807	USS Barnett
6	59982.19723	5.040554523	7.839387785	4.23	26354.10947	630943.4893	USNS Raymond
7	80175.75416	4.988407758	6.104512439	4.04	26748.42842	1068138.074	06039 Jennifer Islands Apt. 443
8	64698.46343	6.025335907	8.147759585	3.41	60828.24909	1502055.817	4759 Daniel Shoals Suite 442
9	78394.33928	6.989779748	6.620477995	2.42	36516.35897	1573936.564	972 Joyce Viaduct
10	59927.66081	5.36212557	6.393120981	2.3	29387.396	798869.5328	USS Gilbert
11	81885.92718	4.42367179	8.167688003	6.1	40149.96575	1545154.813	Unit 9446 Box 0958
12	80527.47208	8.093512681	5.0427468	4.1	47224.35984	1707045.722	6368 John Motorway Suite 700
13	50593.6955	4.496512793	7.467627404	4.49	34343.99189	663732.3969	911 Castillo Park Apt. 717
14	39033.80924	7.671755373	7.250029317	3.1	39220.36147	1042814.098	209 Natasha Stream Suite 961
15	73163.66344	6.919534825	5.993187901	2.27	32326.12314	1291331.518	829 Welch Track Apt. 992
16	69391.38018	5.344776177	8.406417715	4.37	35521.29403	1402818.21	PSC 5330, Box 4420
17	73091.86675	5.443156467	8.517512711	4.01	23929.52405	1306674.66	2278 Shannon View
18	79706.96306	5.067889591	8.219771123	3.12	39717.81358	1556786.6	064 Hayley Unions
19	61929.07702	4.788550242	5.097009554	4.3	24595.9015	528485.2467	5498 Rachel Locks
20	63508.1943	5.94716514	7.187773835	5.12	35719.65305	1019425.937	Unit 7424 Box 2786
21	62085.2764	5.739410844	7.091808104	5.49	44922.1067	1030591.429	19696 Benjamin Cape
22	86294.99909	6.62745694	8.011897853	4.07	47560.77534	2146925.34	030 Larry Park Suite 665
23	60835.08998	5.551221592	6.517175038	2.1	45574.74166	929247.5995	USNS Brown
24	64490.65027	4.21032287	5.478087731	4.31	40358.96011	718887.2315	95198 Ortiz Key
25	60697.35154	6.170484091	7.150536572	6.34	28140.96709	743999.8192	9003 Jay Plains Suite 838
26	59748.85549	5.339339881	7.748681606	4.23	27809.98654	895737.1334	24282 Paul Valley
27	56974.47654	8.287562194	7.312879971	4.33	40694.86951	1453974.506	61938 Brady Falls
28	82173.62608	4.018524685	6.992698757	2.03	38853.91807	1125692.507	3599 Ramirez Springs

Data cleaning and preprocessing:

Data Sources: Gathering comprehensive datasets from real estate listings, including features such as square footage, location, number of bedrooms, and more.

Data Cleaning: Removing missing values, handling outliers, and encoding categorical variables for compatibility with machine learning algorithms.

Feature Selection and Engineering:

Identifying Relevant Features: Utilizing domain knowledge and statistical methods to select the most influential features for price prediction.

Feature Engineering: Creating new features or transforming existing ones to capture hidden patterns in the data.

Advanced regression techniques:

Ridge Regression:

Overview: Introducing L2 regularization to linear regression to mitigate multicollinearity and overfitting.

Hyperparameter Tuning: Optimizing the regularization strength (alpha) to strike a balance between bias and variance.

Lasso Regression:

Overview: Incorporating L1 regularization to linear regression, which encourages feature selection and sparsity in the model.

Hyperparameter Tuning: Tuning the regularization strength (alpha) for feature selection and model performance.

Elastic Net Regression:

Overview: Combining L1 and L2 regularization to benefit from the advantages of both Ridge and Lasso regression.

Hyperparameter Tuning: Finding the optimal combination of L1 and L2 penalties for enhanced model flexibility.

Polynomial Regression:

Overview: Modeling non-linear relationships by introducing polynomial features into the regression equation.

Feature Degree Selection: Determining the appropriate degree of polynomial features to avoid overfitting.

Support Vector Regression (SVR):

Overview: Using Support Vector Machines (SVM) principles for regression tasks, allowing for complex, non-linear relationships.

Kernel Functions: Selecting the right kernel (linear, polynomial, radial basis function, etc.) for the SVR model.

Random Forest Regression:

Overview: Leveraging an ensemble of decision trees to capture complex interactions and improve predictive accuracy.

Tree Depth and Ensemble Size: Tuning the depth of individual trees and the number of trees in the ensemble for optimal results.

Gradient Boosting Regression:

Overview: Building an ensemble of weak learners (typically decision trees) in a sequential manner to improve predictive performance.

Hyperparameter Tuning: Tuning learning rate, tree depth, and ensemble size to achieve the best results.

Neural Networks for Regression:

Overview: Employing deep learning techniques with neural networks to capture complex patterns and non-linear relationships in the data.

Architecture Design: Configuring the neural network architecture, including the number of layers, units, and activation functions.

Model Selection:

Regression Algorithms: Choosing appropriate regression models such as linear regression, decision trees, random forests, support vector regression, or gradient boosting.

Model Evaluation: Employing metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) to evaluate model performance.

Model Training and Validation:

Data Splitting: Dividing the dataset into training and validation sets to assess model generalization.

Hyperparameter Tuning: Optimizing model hyperparameters through techniques like grid search or randomized search.

Model Deployment:

Integrating the trained model into a web application or API for end-users to access and use.

Continuous Monitoring: Ensuring the model's accuracy by regularly updating it with new data and retraining if necessary.

Interpretability:

Explaining Predictions: Employing techniques like SHAP (SHapley Additive exPlanations) values or feature importance scores to interpret how individual features impact house price predictions.

Deployment and prediction:

Deploy the chosen regression model to predict house prices.

Develop a user-friendly interface for users to input property features and receive price predictions.

Program:

House price prediction

Importing Dependencies

```
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")
```

```
/opt/conda/lib/python3.10/site-packages/scipy/_init_.py:146: UserWarning: A NumPy
version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version
1.23.5
warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

Loading Dataset

```
dataset = pd.read_csv('E:/USA_Housing.csv')
```

Model 1-Linear Regression

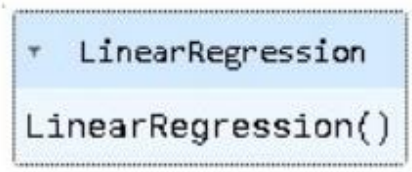
In [1]:

```
model_lr=LinearRegression()
```

In [2]:

```
model_lr.fit(X_train_scal, Y_train)
```

Out[2]:



Predicting Prices

In [3]:

```
Prediction1 = model_lr.predict(X_test_scal)
```

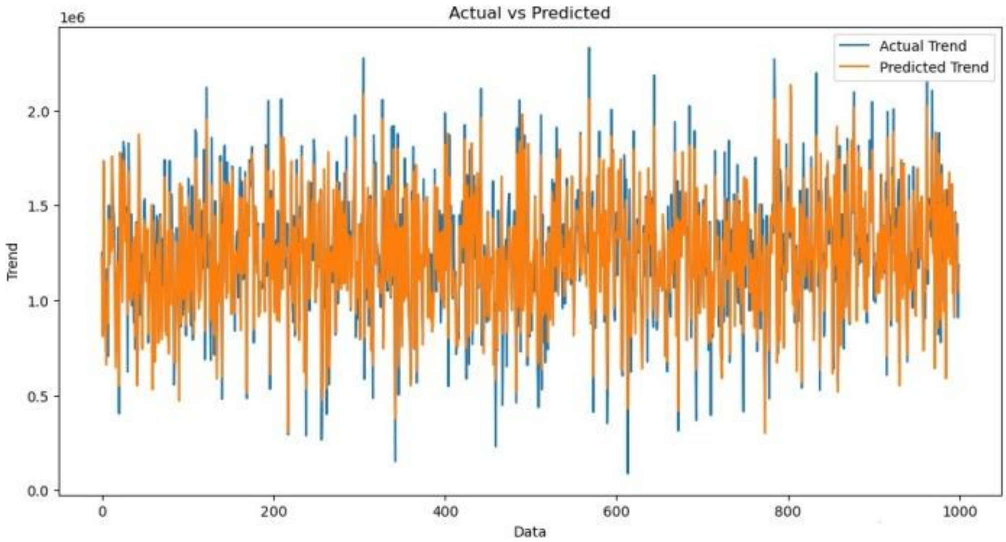
Evaluation of Predicted Data

In [4]:

```
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')
```

Out[4]:

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

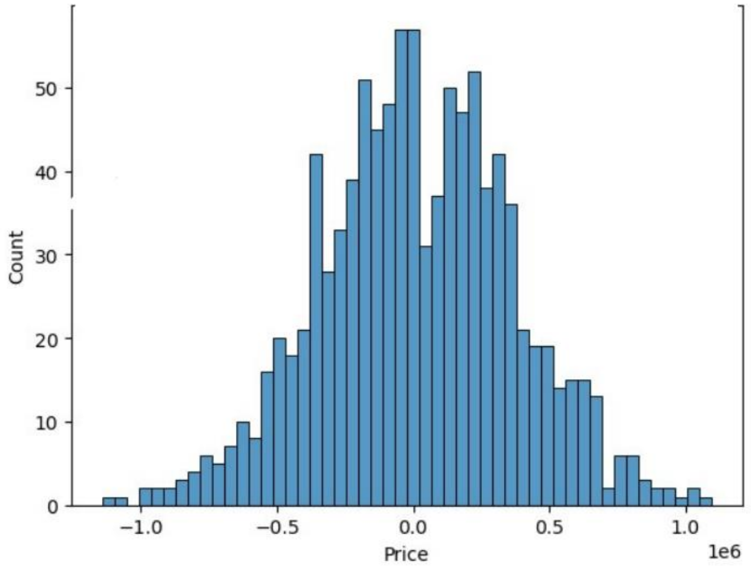


In [5]:

```
sns.histplot((Y_test-Prediction1), bins=50)
```

Out[5]:

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



In [6]:

```
print(r2_score(Y_test, Prediction1))
print(mean_absolute_error(Y_test, Prediction1))
print(mean_squared_error(Y_test, Prediction1))
```

Out[6]:

0.9182928179392918

82295.49779231755

10469084772.975

Model 2 - Support Vector Regressor

In [7]:

```
model_svr = SVR()
```

In [8]:

```
model_svr.fit(X_train_scal, Y_train)
```

Out[8]:

Predicting Prices

In [9]:

```
Prediction2 = model_svr.predict(X_test_scal)
```

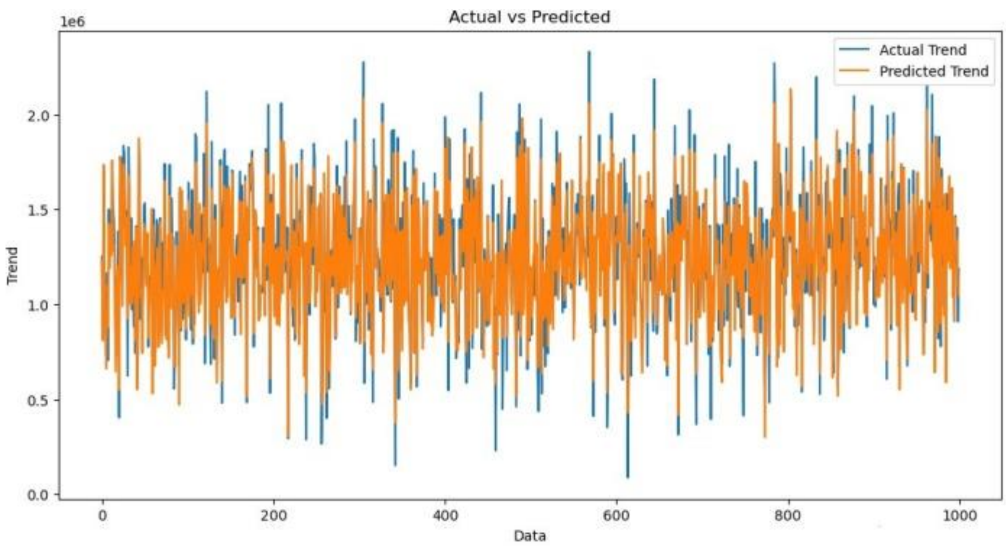
Evaluation of Predicted data

In [10]:

```
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')
```

Out[10]:

Text(0.5, 1.0, 'Actual vs Predicted)

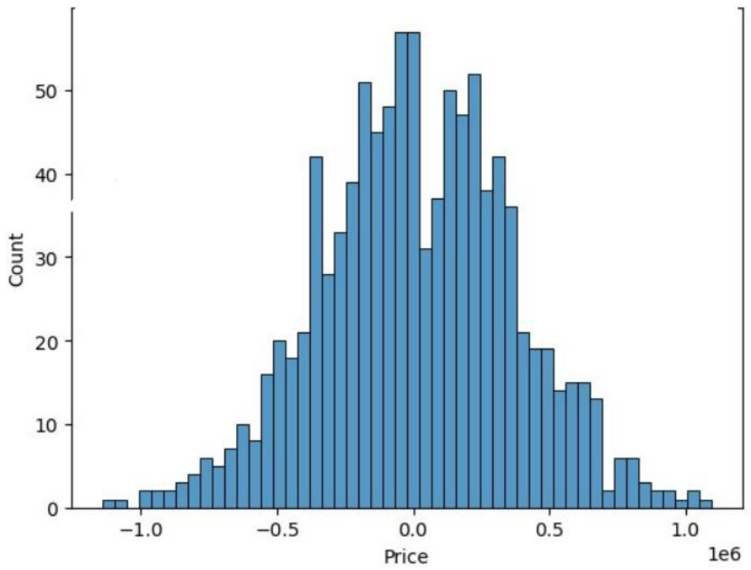


In [11]:

```
sns.histplot((Y_test-Prediction2), bins=50)
```

Out[12]:

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



In [12]:

```
print(r2_score(Y_test, Prediction2))
print(mean_absolute_error(Y_test, Prediction2))
print(mean_squared_error(Y_test, Prediction2))
```

-0.0006222175925689744

286137.81086908665

128209033251.4034

<u>Model 3 - Lasso Regression</u>

In [13]:

```
model_lar = Lasso(alpha=1)
```

In [14]:

```
model_lar.fit(X_train_scal,Y_train)
```

Out[14]:

Predicting Prices

In [15]:

```
Prediction3 = model_lar.predict(X_test_scal)
```

Evaluation of Predicted Data

In [16]:

```
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')

Out[16]:

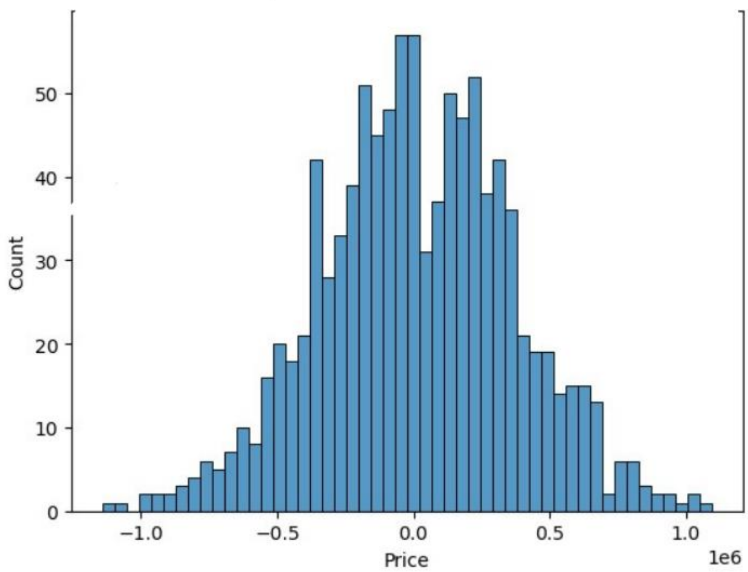
Text(0.5, 1.0, 'Actual vs Predicted')

In [17]:

sns.histplot((Y_test-Prediction3), bins=50)

Out[17]:

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



In [18]:

print(r2_score(Y_test, Prediction2))

print(mean_absolute_error(Y_test, Prediction2))

print(mean_squared_error(Y_test, Prediction2))

-0.0006222175925689744

286137.81086908665

128209033251.4034

<u>Model 4 - Random Forest Regressor</u>

In [19]:

```
model_rf = RandomForestRegressor(n_estimators=50)
```

In [20]:

```
model_rf.fit(X_train_scal, Y_train)
```

Out[20]:

```
RandomForestRegressor
RandomForestRegressor(n_estimators=50)
```

Predicting Prices

In [21]:

```
Prediction4 = model_rf.predict(X_test_scal)
```

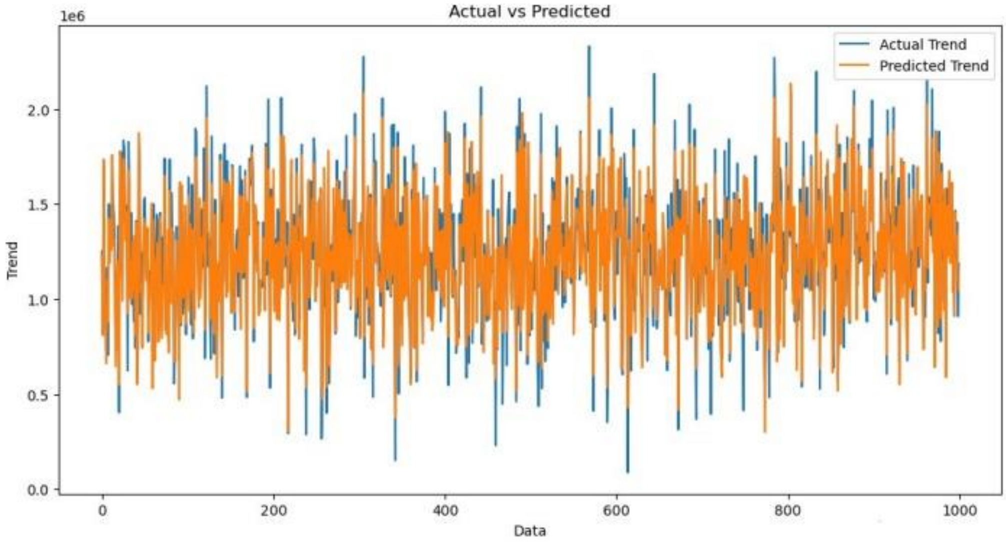
Evaluation of Predicted Data

In [22]:

```
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')
```

Out[22]:

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

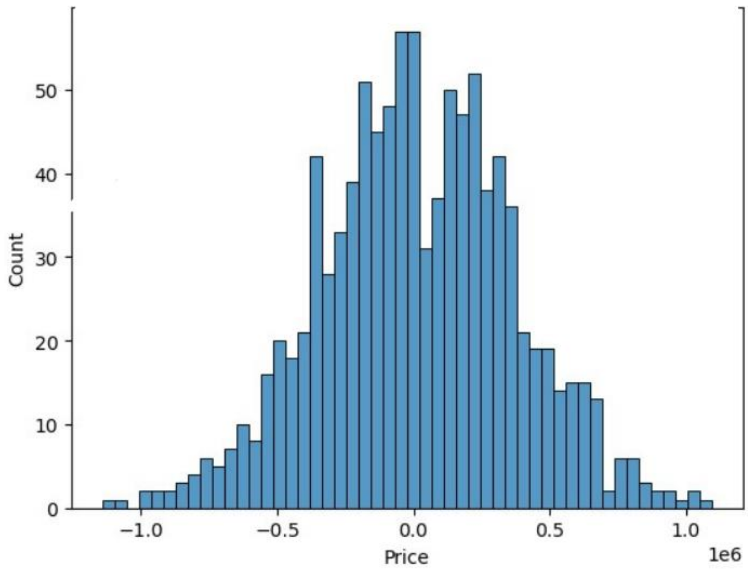


In [23]:

sns.histplot((Y_test-Prediction4), bins=50)

Out[23]:

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



In [24]:

print(r2_score(Y_test, Prediction2))

print(mean_absolute_error(Y_test, Prediction2))

print(mean_squared_error(Y_test, Prediction2))

Out [24] :

-0.0006222175925689744

286137.81086908665

128209033251.4034

Model 5 - XGboost Regressor

In [25]:

```
model_xg = xg.XGBRegressor()
```

In [26]:

```
model_xg.fit(X_train_scal, Y_train)
```

Out[26]:

XGBRegressor

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, ...)

Predicting Prices

In [27]:

```
Prediction5 = model_xg.predict(X_test_scal)
```

Evaluation of Predicted data

In [28]:

```
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')
```

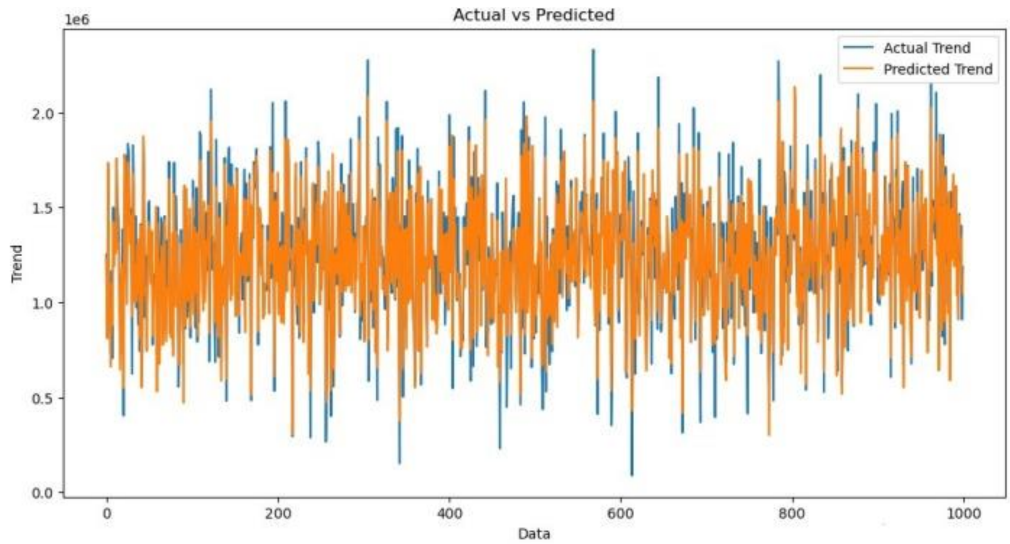
plt.ylabel('Trend')

plt.legend()

plt.title('Actual vs Predicted')

Out[28]:

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

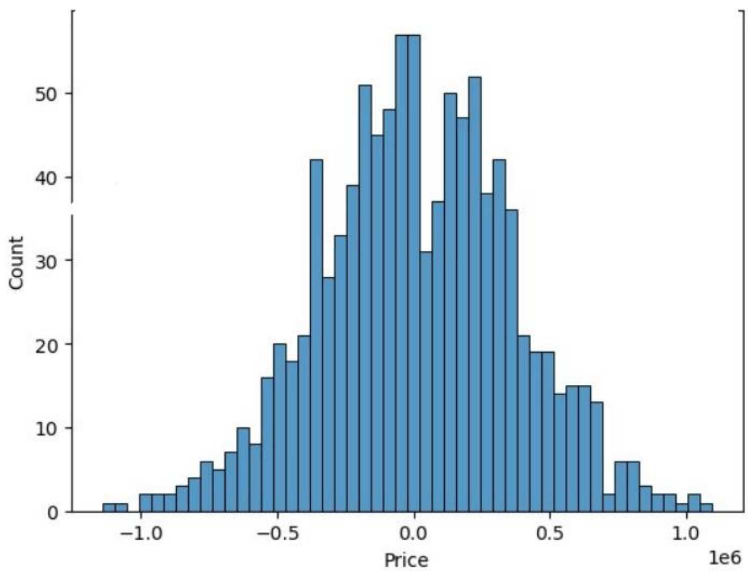


In [29]:

sns.histplot((Y_test-Prediction4), bins=50)

Out[29]:

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



In [30]:

print(r2_score(Y_test, Prediction2))

```
print(mean_absolute_error(Y_test, Prediction2))
print(mean_squared_error(Y_test, Prediction2))
```

Out [30] :

-0.0006222175925689744

286137.81086908665

128209033251.4034

Conclusion and Future Work (Phase 2):

Project Conclusion:

 In the Phase 2 conclusion, we will summarize the key findings and insights from the advanced regression techniques. We will reiterate the impact of these techniques on improving the accuracy and robustness of house price predictions.

 Future Work: We will discuss potential avenues for future work, such as incorporating additional data sources (e.g., real-time economic indicators), exploring deep learning models for prediction, or expanding the project into a web application with more features and interactivity.