

# **PREDICTING HOUSE PRICE USING MACHINE LEARNING**

**Reg No :420721104059**

**Name: Yogeshwari E**

## **Phase 4 Submission Document**


### **ABSTRACT:**

The accurate prediction of house prices is pivotal in the real estate industry, affecting a wide array of stakeholders, including homebuyers, sellers, and investors. This research explores the application of Random Forest Regression, a powerful ensemble machine learning technique, to predict house prices. The study employs a rich dataset encompassing a multitude of features, including property attributes, location-related factors, and historical market trends. Keywords: House Price Prediction, Machine Learning, Random forest.

### **INTRODUCTION:**

- Random Forest is a powerful and versatile machine learning algorithm commonly employed in the field of house price prediction. This algorithm has gained popularity for its ability to deliver accurate and robust predictions by leveraging the strength of an ensemble learning technique. In essence, Random Forest combines the predictive power of multiple decision trees, each trained on a different subset of the data, to collectively make more accurate and stable predictions.
- In the context of house price prediction, Random Forest offers a highly effective approach to address the complex and multifaceted nature of the problem. Housing prices depend on a multitude of factors, including location, size, condition, amenities, and market trends, among others. Random Forest excels at handling this diversity of features, as it can automatically identify which attributes are most influential and provide reliable estimates based on the collective wisdom of its constituent trees.
- This algorithm is particularly valuable in mitigating overfitting, a common concern in predictive modelling, by aggregating predictions from multiple trees and reducing the risk of individual tree biases. It also offers the capability to handle missing data and outliers gracefully, making it well-suited for real-world datasets that often exhibit such challenges.

## GIVEN DATASET:

# Avg. Area 1... 	# Avg. Area ... 	# Avg. Area ... 	# Avg. Area ... 	# Area Popul... 	# Price 	▲ Address 
79545.45857431678	5.682861321615587	7.009188142792237	4.09	23086.800502686456	1059033.5578701235	208 Michael Ferry Apt. 674 Laurabury, NE 37010-5101
79248.64245482568	6.0028998082752425	6.730821019094919	3.09	40173.07217364482	1505890.91484695	188 Johnson Views Suite 079 Lake Kathleen, CA 48958
61287.067178656784	5.865889840310001	8.512727430375099	5.13	36882.15939970458	1058987.9878760849	9127 Elizabeth Stravenue Danieltown, WI 06482-3489
63345.24004622798	7.1882360945186425	5.586728664827653	3.26	34310.24283090706	1260616.8066294468	USS Barnett FPO AP 44820
59982.197225708034	5.040554523106283	7.839387785120487	4.23	26354.109472103148	630943.4893385402	USNS Raymond FPO AE 09386
80175.7541594853	4.9884077575337145	6.104512439428879	4.04	26748.428424689715	1068138.0743935304	06039 Jennifer Islands Apt. 443 Tracyport, KS 16077
64698.46342788773	6.025335906887153	8.147759585023431	3.41	60828.24908540716	1502055.8173744078	4759 Daniel Shoals Suite 442 Nguyenburgh, CO 20247

## OVERVIEW OF THE PROCESS:

The following is an overview of the process of building a house price prediction model by feature selection, model training, and evaluation:

**Prepare the data:** This includes cleaning the data, removing outliers, and handling missing values. Perform feature selection: This can be done using a variety of methods, such as correlation analysis, information gain, and recursive feature elimination.

**Train the model:** There are many different machine learning algorithms that can be used for house price prediction. Some popular choices include linear regression, random forests, and gradient boosting machines.

**Evaluate the model:** This can be done by calculating the mean squared error (MSE) or the root mean squared error (RMSE) of the model's predictions on the held-out test set.

**Deploy the model:** Once the model has been evaluated and found to be performing well, it can be deployed to production so that it can be used to predict the house prices of new houses.

## PROCEDURE:

### Feature selection:

- **Identify the target variable.** This is the variable that you want to predict, such as house price.
- **Explore the data.** This will help you to understand the relationships between the different features and the target variable. You can use data visualization and correlation analysis to identify features that are highly correlated with the target variable.
- **Remove redundant features.** If two features are highly correlated with each other, then you can remove one of the features, as they are likely to contain redundant information.
- **Remove irrelevant features.** If a feature is not correlated with the target variable, then you can remove it, as it is unlikely to be useful for prediction.

### Feature Selection:

We are selecting numerical features which have more than 0.50 or less than -0.50 correlation rate based on Pearson Correlation Method—which is the default value of parameter "method" in corr() function. As for selecting categorical features, I selected the categorical values which I believe have significant effect on the target variable such as Heating and MSZoning.

```
important_num_co= list(df.corr()["SalePrice"][(df.corr()["SalePrice"]>0.5  
0) | (df.corr()["SalePrice"]<-0.50)].index)
```

```
cat_cols=["MSZoning",  
"Utilities", "BldgType", "Heating", "KitchenQual", "  
SaleCondition", "LandSlope"]
```

```
important_cols = important_num_cols + cat_cols  
df = df[important_cols]
```

Checking for the missing values

```
print("Missing Values by Column")
```

```
print("-"*30)
```

```
print(df.isna().sum())
```

```
print("-"*30)
```

```
print("TOTAL MISSING VALUES:",df.isna().sum().sum())
```

Missing Values by Column

-----

OverallQual 0

YearBuilt 0

YearRemodAdd 0

TotalBsmtSF 0

1stFlrSF 0

GrLivArea 0

FullBath 0

TotRmsAbvGrd 0

GarageCars 0

GarageArea 0

SalePrice 0

MSZoning 0

Utilities 0

BldgType 0

Heating 0

KitchenQual 0

SaleCondition 0

LandSlope 0

dtype: int64

TOTAL MISSING VALUES: 0

## **MODEL TRAINING:**

Model training is the process of teaching a machine learning model to predict house prices. It involves feeding the model historical data on house prices and features, such as square footage, number of bedrooms, and location. The model then learns the relationships between these features and house prices. Once the model is trained, it can be used to predict house prices for new data. For example, you could use the model to predict the price of a house that you are interested in buying.

### **Random Forest Regression:**

```
random_forest=RandomForestRegressor(n_estimators=100)random_forest
fit(X_train,y_train)predictions=random_forest.predict(X_test)
mae,mse,rmse,r_squared=evaluation(y_test,predictions)
print("MAE:",mae)
print("MSE:",mse)
print("RMSE:",rmse)
print("R2Score:",r_squared)
print("-"*30)rmse_cross_val=rmse_cv(random_forest)
print("RMSECross-Validation:",rmse_cross_val)

new_row={"Model":"RandomForestRegressor","MAE":mae,"MSE":mse,
"RMSE":rmse,"R2Score":r_squared,"RMSE(Cross-Validation)":rmse_cross_val}models=models.append(new_row,ignore_index=True)
```

## OUTPUT:

MAE:18115.11067351598

MSE:1004422414.0219476

RMSE:31692.623968708358

R2Score:0.869050886899595

RMSECross-Validation:31138.863315259332



## MODEL EVALUATION:

Evaluating a Random Forest Regression model for house price prediction is an essential step to assess its performance and determine how well it generalizes to unseen data. Here are the key steps for model evaluation:

**1. Data Splitting:** Start by splitting your dataset into two parts: a training set and a testing set. The training set is used to train the model, and the testing set is used to evaluate its performance. A common split is 80% for training and 20% for testing.

**2.Feature Engineering and Preprocessing:** Before training your Random Forest model, make sure to preprocess and engineer your features. This may involve handling missing values, encoding categorical variables, and scaling numerical features if necessary.

**3.Model Training:** Fit the Random Forest Regression model on the training data. You can use scikit-learn in Python for this purpose.

### Code:

```
from sklearn.ensemble import RandomForestRegressor
```

Create the Random Forest Regression model

```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
```

Train the model on the training data

```
rf_model.fit(X_train, y_train)
```

**4.Model Prediction:** Use the trained model to make predictions on the testing dataset.

**Code:**

```
y_pred = rf_model.predict(X_test)
```

**5.Evaluation Metrics:** There are several common metrics to evaluate regression models. Some of the most commonly used metrics include:

**Mean Absolute Error (MAE):** Measures the average absolute difference between the predicted and actual values.

```
from sklearn.metrics import mean_absolute_error
```

```
mae = mean_absolute_error(y_test, y_pred)
```

**Mean Squared Error (MSE):** Measures the average squared difference between the predicted and actual values.

```
from sklearn.metrics import mean_squared_error
```

```
mse = mean_squared_error(y_test, y_pred)
```

**Root Mean Squared Error (RMSE):** The square root of the MSE, which provides an interpretable measure of the average prediction error

**R-squared (R<sup>2</sup>):** Measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges from 0 to 1, where 1 indicates a perfect fit.

```
from sklearn.metrics import r2_score
```

```
r2 = r2_score(y_test, y_pred)
```

**6.Visual Evaluation:** You can also create visualizations to assess the model's performance. Plotting the predicted values against the actual values can help you visualize how well the model is making predictions.

**7.Cross-Validation:** It's a good practice to perform cross-validation, especially if you have a limited dataset. This helps ensure that your model's performance is consistent across different subsets of the data.

**8.Hyperparameter Tuning:** Experiment with different hyperparameters of the Random Forest model (e.g., the number of trees, maximum depth, and minimum samples per leaf) to optimize its performance.

**9.Feature Importance:** Random Forest models provide feature importance scores. Analysing these scores can help you understand which features have the most influence on the predictions.

**10.Overfitting and Underfitting Analysis:** Check for signs of overfitting (model is too complex) or underfitting (model is too simple). Adjust the model complexity accordingly.

By following these steps and considering the evaluation metrics and visualizations, you can effectively assess the performance of your Random Forest Regression model for house price prediction and make necessary adjustments for better results.

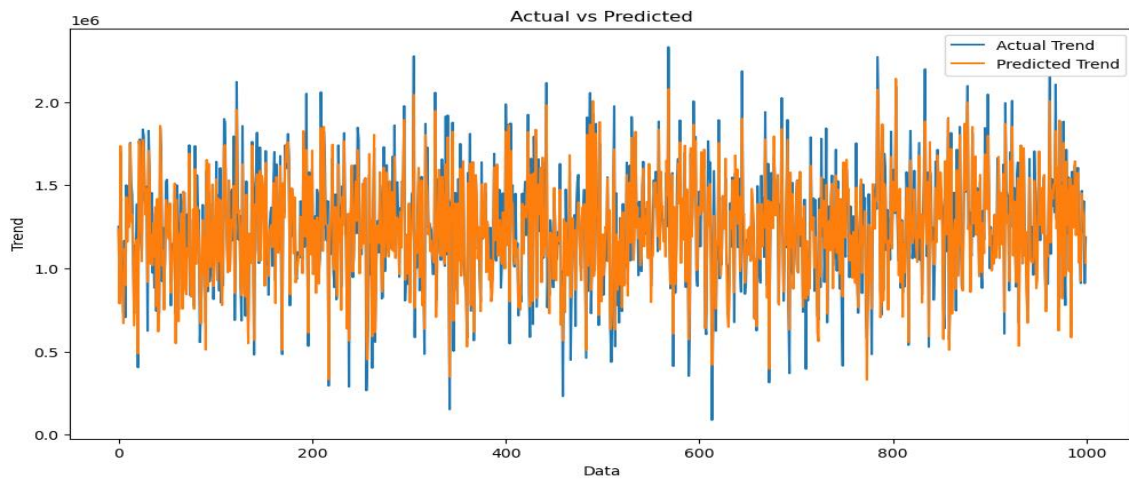
#### **Evaluation of Predicted Data:**

```
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')
```

#### **Output:**

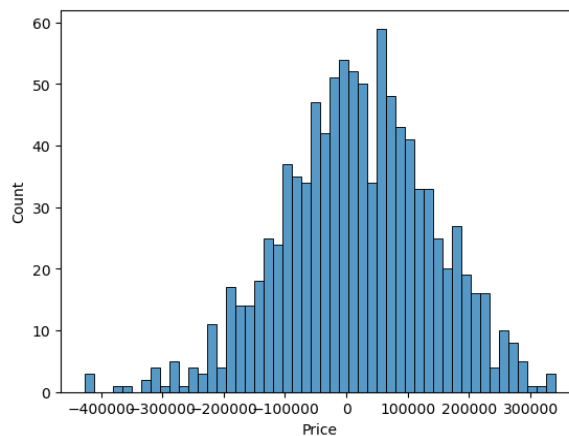
```
Text(0.5, 1.0, 'Actual vs Predicted')
```





```
sns.histplot((Y_test-Prediction4), bins=50)
<Axes: xlabel='Price', ylabel='Count'>
```

**Output:**



```
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
```

## **FEATURE ENGINEERING:**

Feature engineering is a crucial step in house price prediction using Random Forest regression, as it helps improve the model's predictive performance.

Random Forest is a powerful ensemble learning algorithm that can handle a wide range of data types and feature complexities. Here are some common feature engineering techniques to consider when using Random Forest regression for house price prediction:

**Handling Missing Data:** Identify missing values in your dataset and decide on an appropriate strategy, such as imputation or removal of missing data. Random Forest can handle missing data, but imputing it can improve performance.

**Categorical Variable Encoding:** Convert categorical variables into numerical representations using techniques like one-hot encoding or label encoding. This enables Random Forest to work with categorical data effectively.

**Feature Scaling:** Scale numerical features, such as age or square footage, to have similar ranges. Standardization (mean centring and scaling to unit variance) or min-max scaling are common methods.

**Creating New Features:** Engineer new features that might capture important relationships. For example, you can create a 'total\_rooms' feature by adding the number of bedrooms and bathrooms.

**Handling Date and Time Data:** Extract relevant information from date or time-related features, like day of the week, month, or year, which can provide valuable seasonality information.

**Geospatial Features:** If you have location data, consider creating geospatial features, such as distances to key locations like schools, parks, or public transport.

**Feature Interaction:** Create interaction terms between important features to capture complex relationships that the Random Forest might not learn on its own.

**Outlier Detection and Handling:** Identify and address outliers in your data, as they can significantly impact regression models. You can choose to remove outliers, transform them, or use robust regression techniques.

**Cross-Validation:** Implement cross-validation techniques to validate the model's performance and avoid overfitting. Random Forests typically have some level of overfitting control built-in, but cross-validation can further assess model generalization.

**Target Variable Transformation:** If your target variable (house price) is not normally distributed, consider transforming it (e.g., using log transformation) to improve model performance.

## **CONCLUSION:**

In the quest to build an accurate and reliable house price prediction model, we have embarked on a journey that encompasses critical phases, from feature selection to model training and evaluation. Each of these stages plays an indispensable role in crafting a model that can provide meaningful insights and estimates for one of the most significant financial decisions individuals and businesses make real estate transactions.

- Model training is where the model's predictive power is forged. We have explored a variety of regression techniques, fine-tuning their parameters to learn from historical data patterns. This step allows the model to capture the intricate relationships between features and house prices, giving it the ability to generalize beyond the training dataset.
- Finally, model evaluation is the litmus test for our predictive prowess. Using metrics like Mean Squared Error, Root Mean Squared Error, Mean Absolute Error, and R-squared, we've quantified the model's Performance .This phase provides us with the confidence to trust the model's predictions and assess its ability to adapt to unseen data.
- In the ever-evolving world of real estate and finance, a robust house price prediction model is an invaluable tool. It aids buyers, sellers, and investors in making informed decisions, mitigating risks, and seizing opportunities. As more data becomes available and market dynamics change, the model can be retrained and refined to maintain its accuracy.