# House Price Prediction using XGBoostRegressor

## Overview

The **House Price Prediction** project aims to predict the prices of houses based on various features such as location, size, number of rooms, and other factors. The model is trained using the **XGBoostRegressor**, a powerful machine learning algorithm optimized for structured data.

## Dataset

The dataset contains various features related to houses, including:

* **Id**: Unique identifier for each house.
* **LotArea**: Size of the lot in square feet.
* **YearBuilt**: Year the house was constructed.
* **OverallQual**: Overall quality rating of the house.
* **TotalBsmtSF**: Total square feet of basement area.
* **GrLivArea**: Above-ground living area in square feet.
* **GarageCars**: Number of cars that fit in the garage.
* **SalePrice**: The target variable representing the selling price of the house.

## Data Preprocessing

Before training the model, the following preprocessing steps were performed:

1. **Handling Missing Values**: Imputed missing values using mean/median for numerical features and mode for categorical features.
2. **Encoding Categorical Variables**: Applied one-hot encoding for categorical variables.
3. **Feature Scaling**: Standardized numerical features using Min-Max Scaling.
4. **Feature Selection**: Selected important features using correlation analysis.
5. **Splitting Data**: Split the dataset into training (80%) and testing (20%) sets.

## Model Training

The **XGBoostRegressor** model was trained using the following steps:

* Imported the XGBRegressor from xgboost.
* Initialized the model with optimized hyperparameters:
* from xgboost import XGBRegressor

model = XGBRegressor(n\_estimators=500, learning\_rate=0.05, max\_depth=6, random\_state=42)

* Fit the model to the training data:
* Model.fit(x\_train,y\_train)

### Why Use **XGBoost** for House Price Prediction?

**XGBoost (Extreme Gradient Boosting)** is a powerful machine learning algorithm that is highly efficient for structured/tabular data. Here’s why it was chosen for this problem:

### 1. **High Predictive Performance**

* XGBoost often outperforms traditional regression models (like Linear Regression or Decision Trees) by capturing complex patterns in data.
* It reduces both bias and variance by using **gradient boosting**, making it well-suited for numerical predictions like house prices.

### 2. **Handles Missing Data Well**

* Real-world datasets often have missing values, and XGBoost can **automatically handle missing values**, unlike some models that require extensive preprocessing.

### 3. **Feature Importance & Selection**

* XGBoost ranks **feature importance**, allowing us to identify the most influential features (e.g., lot size, number of rooms) in predicting house prices.
* This helps in better feature engineering and model interpretability.

### 4. **Handles Non-Linear Relationships**

* Unlike Linear Regression, XGBoost can model **non-linear** relationships between features and the target variable, making it more effective for real estate data.

### 5. **Regularization to Prevent Overfitting**

* XGBoost includes **L1 (Lasso) and L2 (Ridge) regularization**, which helps prevent overfitting and ensures better generalization to unseen data.

### 6. **Scalability & Speed**

* It is highly optimized for speed and memory efficiency, making it **faster** than traditional boosting algorithms.
* XGBoost can be parallelized, making it suitable for large datasets.

### 7. **Hyperparameter Tuning for Optimization**

* XGBoost provides flexible tuning options, allowing us to adjust:
  + n\_estimators (number of trees)
  + max\_depth (tree depth)
  + learning\_rate (step size for updating weights)
  + subsample (fraction of data used per boosting round)
* This helps in **optimizing performance** and avoiding overfitting.

## Model Evaluation

The model was evaluated using the following metrics:

* **Mean Absolute Error (MAE)**: Measures average absolute error.
* **Root Mean Squared Error (RMSE)**: Measures error magnitude.
* **R-Squared (R²)**: Indicates how well the model explains the variance in the data.

Example evaluation code:

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)

r2 = r2\_score(y\_test, y\_pred)

print(f"MAE: {mae}, RMSE: {rmse}, R²: {r2}")

## Results

* **MAE**: 17000
* **RMSE**: 24500
* **R²**: 0.89

## Conclusion

The **XGBoostRegressor** model effectively predicts house prices with high accuracy. Future improvements can include hyperparameter tuning, feature engineering, and using ensemble learning techniques for better performance.

