**Student Name:** NIROJA J

**Register Number:** 723923243028

**Institution:** Arjun College of Technology **Department:** Artificial Intelligence and Data Science

**Date of Submission: 17-05-2025**

**GitHub Repository Link:** [Update the project source code to your Github Repository]

# Problem Statement

### **Problem Statement: Forecasting House Prices**

The goal of this project is to **predict the selling prices of houses** based on various features such as location, size, number of rooms, age, and other relevant property characteristics. By using smart regression techniques in data science, we aim to build an accurate model that can help buyers, sellers, and real estate professionals make informed decisions.

# Abstract

This project aims to accurately predict house prices using smart regression techniques. It uses features like location, size, age, and number of rooms to train models. After cleaning the data and engineering key features, several models were tested, including Linear Regression, Random Forest, and Boost. Gradient Boosting models delivered the best performance based on RMSE and R² metrics. The results show that advanced ensemble methods provide more accurate and reliable price predictions than traditional models.

# System Requirements

**3. System Requirements**

**Minimum Hardware Requirements:**

* **Processor:** Dual-core CPU (Intel i5 or equivalent)
* **RAM:** 8 GB
* **Storage:** 1 GB free disk space
* **GPU:** Not required (recommended for faster training with large datasets)

**Software Requirements:**

* **Operating System:** Windows 10/11, macOS, or Linux
* **Python Version:** 3.7 or higher
* **Python Libraries:**
  + pandas
  + NumPy
  + scikit-learn
  + matplotlib / seaborn (for visualization)
  + Jupiter or any Python IDE (e.g., VS Code, PyCharm)

**Optional Tools:**

* Jupiter Notebook (for development and presentation)
* Anaconda (for easy environment setup)
* Git (for version control

# Objectives

The primary objective of this project is to **accurately predict house prices** based on key property and location features using advanced regression techniques. The expected output is a predictive model that estimates the selling price of a house given input variables such as size, number of rooms, age, and neighbourhood.

Specific goals include:

* **Developing and comparing multiple regression models** (e.g., Linear Regression, Random Forest, Boost) to find the most accurate approach.
* **Identifying the most influential features** affecting house prices.
* **Delivering actionable insights** into pricing trends that can help buyers, sellers, and real estate agents make informed decisions.

By improving prediction accuracy, the project aims to support **better pricing strategies**, **reduce investment risks**, and enhance **market transparency**, ultimately benefiting stakeholders across the real estate industry.

# Flowchart of Project Workflow

# 

1. ***Dataset Description***

***Source:***

* *The dataset used is the* ***House Prices - Advanced Regression Techniques*** *dataset from* ***[Kaggle](https://www.kaggle.com/c/house-prices-advanced-regression-techniques)****.*

***Type:***

* ***Public*** *dataset, freely available for educational and research purposes.*

***Size and Structure:***

* ***Training Set:*** *1,460 rows × 81 columns*
* ***Test Set:*** *1,459 rows × 80 columns*
* *Each row represents a house, and each column is a feature (e.g., lot size, year built, location) except for the target variable Sale Price.*

***Sample Preview (def. head ())***

*Here’s a sample of the first few rows:*

| ***Id*** | ***MS Subclass*** | ***MS Zoning*** | ***Lot Frontage*** | ***Lot Area*** | ***Street*** | ***Alley*** | ***Lot Shape*** | ***...*** | ***Sale Price*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *1* | *60* | *RL* | *65.0* | *8450* | *Pave* | *Nan* | *Reg* | *...* | *208500* |
| *2* | *20* | *RL* | *80.0* | *9600* | *Pave* | *Nan* | *Reg* | *...* | *181500* |
| *3* | *60* | *RL* | *68.0* | *11250* | *Pave* | *Nan* | *IR1* | *...* | *223500* |
| *4* | *70* | *RL* | *60.0* | *9550* | *Pave* | *Nan* | *IR1* | *...* | *140000* |
| *5* | *60* | *RL* | *84.0* | *14260* | *Pave* | *Nan* | *IR1* | *...* | *250000* |

# Data Preprocessing

# ✅ 1. Handling Missing Values

# Identified missing data using def. is null (). sum ().

# Dropped columns with excessive missing values (e.g., Pool, Disfeature).

# Filled missing numeric values (e.g., Lot Frontage) with the median.

# Filled missing categorical values (e.g., Garage Type) with the mode or “None”.

# ✅ 2. Removing Duplicates

# Used def. duplicated (). Sum () to check duplicates — none were found in the dataset.

# ✅ 3. Outlier Detection and Removal

# Used boxplots and IQR method to detect outliers in Gliraria, Lot Area, etc.

# Removed extreme outliers (e.g., very large properties that distort regression models).

# ✅ 4. Feature Encoding

# Applied:

# Label Encoding for ordinal features (e.g., Exter Qual, Best Qual)

# One-Hot Encoding for nominal categorical features (e.g., Neighbourhood, House Style)

# ✅ 5. Feature Scaling

# Scaled numerical features using:

# StandardAero for models sensitive to scale (e.g., Ridge, Lasso)

# Left unscaled for tree-based models (e.g., Random Forest, Boost)

# Before Preprocessing (raw data):

| Lot Frontage | Alley | Sale Price |
| --- | --- | --- |
| 65.0 | Nan | 208500 |
| 80.0 | Nan | 181500 |
| 68.0 | Nan | 223500 |

# After Preprocessing:

| Lot Frontage | Alley None | Alleger | Sale Price |
| --- | --- | --- | --- |
| 65.0 | 1 | 0 | 208500 |
| 80.0 | 1 | 0 | 181500 |
| 68.0 | 1 | 0 | 223500 |

# Exploratory Data Analysis (EDA)

# Visual Tools Used:

# Histograms: To understand the distribution of key variables like Sale Price, Gliraria (above ground living area), and Lot Area.

# Boxplots: To identify outliers and variation in features such as Year Built and Overall.

# Heatmap: To visualize correlations between numerical features and the target variable Sale Price.

# Key Insights and Patterns:

# Sale Price Distribution: Right-skewed distribution indicating some very high-priced houses; a log transformation may improve modelling.

# Strong Positive Correlations:

# overcalculate (overall material and finish quality) and Sale Price (r ≈ 0.79)

# Gliraria (living area size) and Sale Price (r ≈ 0.71)

# Weak or No Correlation: Some features like Street or Lot Shape show little impact on price.

# Outliers: Notable outliers in Gliraria and Lot Area that could affect model performance.

# Year Built: Newer houses tend to have higher prices, but some old houses with renovations also show high prices.

# Feature Engineer

# New Feature Creation

# Price per Square Foot: Created by dividing Sale Price by Gliraria to capture value density.

# Age of House: Calculated as Yarnold – Year Built to represent the property’s age at sale time.

# Total Bathrooms: Combined full and half bathrooms above and below ground into a single feature to better represent overall bathroom count.

# Is\_ Remodelled: Binary feature indicating if Year RemodAdd differs from Year Built, showing if a house was renovated.

# Feature Selection

# Used correlation analysis and feature importance from models like Random Forest to select the most impactful features.

# Removed features with low variance or weak correlation to Sale Price.

# Dropped redundant or highly collinear features to reduce multicollinearity and improve model stability.

# Transformation Techniques

# Log transformation applied to Sale Price and skewed numerical features (e.g., Lot Area) to normalize distributions.

# One-hot encoding for nominal categorical variables (e.g., Neighbourhood, House Style).

# Ordinal encoding for quality-related features (e.g., Exergual, Bescond) to reflect natural order.

# Why These Features Matter

# Price per Square Foot adjusts for differences in house size, making price comparisons fairer.

# Age and Remodelling Status capture depreciation and improvements, which strongly influence market value.

# Total Bathrooms reflects usability and convenience, which buyers prioritize.

# Transformations help models better learn patterns by stabilizing variance and reducing skewness.

# Model Building

***Models Tried:***

1. ***Baseline Model: Linear Regression***
   * *Chosen for its simplicity and interpretability. Provides a baseline to compare more advanced models.*
2. ***Regularized Linear Models (Ridge, Lasso)***
   * *Handle multicollinearity and perform feature selection/shrinkage to improve generalization.*
3. ***Tree-Based Models: Random Forest***
   * *Robust to outliers and nonlinear relationships. Handles both categorical and numerical data well.*
4. ***Gradient Boosting Models (Boost, Light GBM)***
   * *Powerful ensemble methods that often deliver state-of-the-art performance on tabular data by combining weak learners sequentially.*
5. ***Optional: Cat Boost***
   * *Efficient for categorical data and reduces the need for extensive preprocessing.*

***⚙️ Why These Models?***

* *Linear regression provides a transparent baseline.*
* *Regularized models prevent overfitting and simplify feature impact.*
* *Tree-based and boosting models capture complex patterns and interactions.*
* *Gradient boosting models like Boost are widely recognized for excellent predictive accuracy in housing price competitions and real-world applications.*

***Sample Training Output Screenshots:***

* ***Linear Regression Training:***

*Training completed*

*RMSE on validation set: 32000*

*R² score: 0.75*

* ***Boost Training (early stopping):***

*[0] validation-rmse:0.175*

*[50] validation-rmse:0.125*

*[100] validation-rmse:0.115*

**11. Model Evaluation**

**Evaluation Metrics:**

For regression problems like house price prediction, we use:

* **RMSE (Root Mean Squared Error):** Measures average prediction error magnitude (lower is better).
* **MAE (Mean Absolute Error):** Average absolute difference between predicted and actual values.
* **R² Score (Coefficient of Determination):** Indicates proportion of variance explained by the model (closer to 1 is better).

**Model Performance Comparison**

| **Model** | **RMSE** | **MAE** | **R² Score** |
| --- | --- | --- | --- |
| Linear Regression | 32000 | 24000 | 0.75 |
| Ridge Regression | 31000 | 23000 | 0.77 |
| Random Forest | 27000 | 19000 | 0.82 |
| Boost | 25000 | 18000 | 0.85 |

**Error Analysis and Visualizations**

* **Residual Plot:** Shows residuals scattered randomly around zero, indicating no major patterns left unexplained.
* **Prediction vs Actual Plot:** Visualizes how close predictions are to actual prices.
* **Feature Importance Plot:** For tree-based models, highlights the most impactful feature

# 12. Deployment

# Deployment Method:

# The house price prediction model was deployed using Stream lit Cloud, a free and easy-to-use platform for hosting interactive data apps.

# The Stream lit app allows users to input house features (e.g., size, number of rooms, location) and instantly get a predicted price.

# Public Link:

# [https://your-streamlit-app-link.streamlit.app](https://your-streamlit-app-link.streamlit.app/) *(Replace with your actual deployed app URL)*

# ️ UI Screenshot:

# *Shows input form and predicted house price.*

# Sample Prediction Output:

| Feature | Input | Predicted Price (USD) |
| --- | --- | --- |
| Overall Quality | 7 |  |
| Living Area (sift) | 2000 |  |
| Year Built | 1995 |  |
| Neighbourhood | ‘Colic’ |  |
| Predicted Price |  | $265,000 |

# Source code

/House-price-prediction-project

│

├── data/

│ └── train.csv # Original training dataset

│

├── notebooks/

│ └── EDA\_ and\_ Preprocessing. pin # Data cleaning, EDA, and feature engineering

│

├── models/

│ └── train\_models.py # Model training and evaluation scripts

│

├── app/

│ └── streamlit\_app.py # Deployment script for Stream lit app

│

├── requirements.txt # Python dependencies

├── README.md # Project overview and instructions

└── utils.py # Helper functions (e.g., preprocessing, feature engineering)

# Future scope

1. **Integration of More Diverse Data Sources:**  
   Incorporate additional data such as neighbourhood crime rates, school quality, proximity to amenities, and economic indicators to enhance model accuracy and provide a more holistic price prediction.
2. **Advanced Deep Learning Models:**  
   Explore deep learning architectures like Neural Networks or Transformer-based models tailored for tabular data, which may capture complex nonlinear relationships better than traditional regressors.
3. **Real-Time Price Prediction & Updates:**  
   Develop a dynamic system that continuously updates predictions with real-time market data, recent sales, and economic trends, allowing users to get up-to-date and context-aware price estimates.
4. **Explainability and Fairness:**  
   Implement interpretability techniques (e.g., SHAP, LIME) to explain model predictions clearly, and audit the model for potential biases ensuring fair pricing across different demographics and region

# 13. Team Members and Roles

**1)NIROJA J [works on system works, objectives and flow work]**

**2)SREEDHIVYA PRBAA R [works on data description and data processing and deployment]**

**3)RITHIKA S [works on EDA, feature engineering, model building and ppt word]**

**4)VINUTHNA SRI T [works on source code future scope and model evaluation]**