**Student Name: Deepa S**

**Register Number:** 613023104013

**Institution:** Vivekanandha College of Technology for Women **Department:** Computer Science and Engineering

**Date of Submission:** 02.05.2025

**GitHub Repository Link: https://github.com/S-Deepa2006/Project.git**

# Problem Statement

The real-world problem at hand is **accurately forecasting house prices** using advanced regression techniques in data science. With the continuous growth of the real estate sector, reliable and data-driven price prediction models are crucial for buyers, sellers, investors, and real estate professionals. Traditional pricing methods often rely on subjective judgment or outdated valuation practices, which can lead to inconsistencies and mispricing.

**Problem Type:** Supervised regression (To **predict a continuous target variable)**

# Abstract

This project aims to forecast house prices using advanced regression techniques in data science. By leveraging publicly available housing datasets, we preprocess and analyze the data to extract meaningful insights. Various regression models such as Linear Regression, Random Forest, and Gradient Boosting are applied and compared. The best-performing model is deployed using Streamlit to enable user interaction. This project demonstrates how smart regression methods can significantly improve the accuracy and reliability of housing price predictions.

# System Requirements

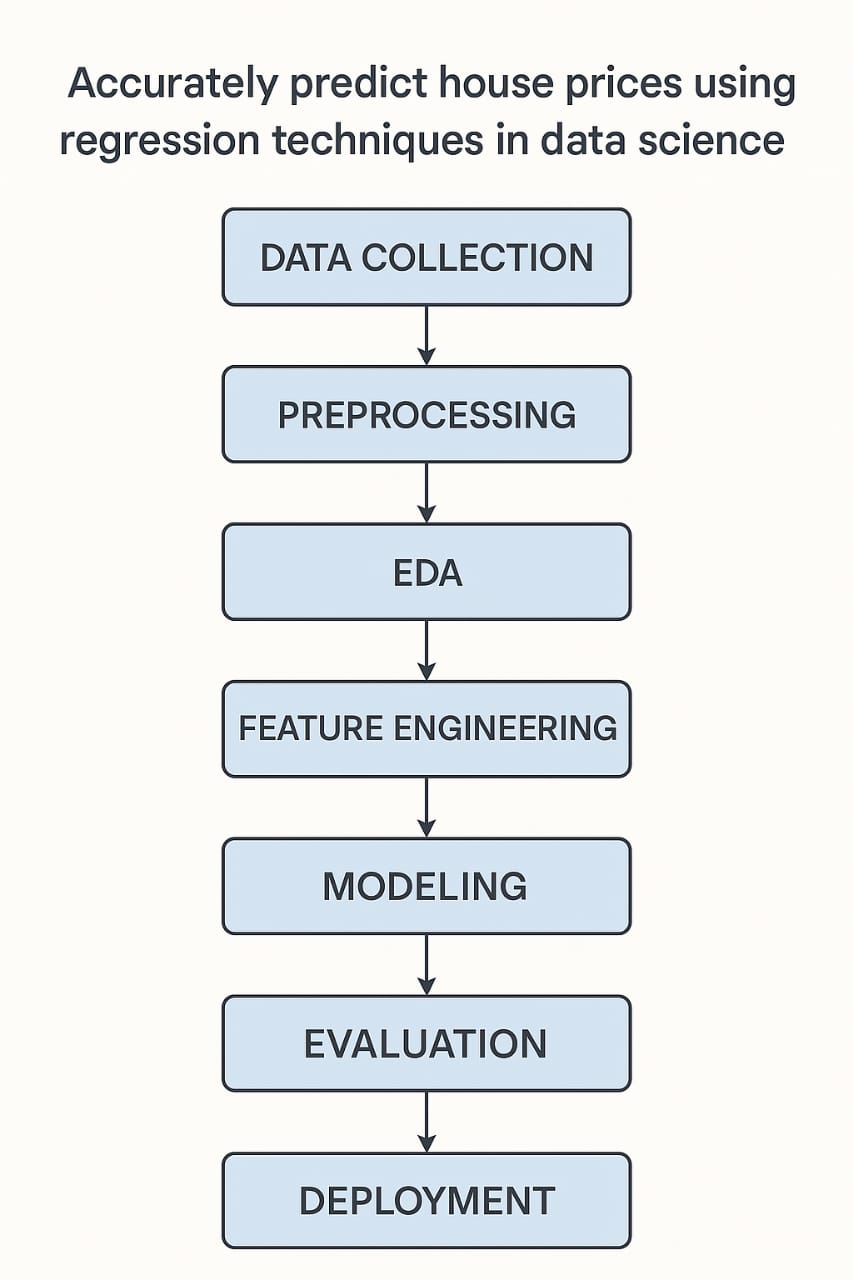
* ***Hardware****:*
* **RAM:** Minimum 8 GB
* **Processor:** Intel Core i5
* **Software:**
* **Python Version:** Python 3.8
* **Libraries:** numpy, pandas, matplotlib, seaborn, scikit-learn, xgboost.
* **IDE:** Jupyter Notebook ,Google Colab

# Objectives

* Predict housing prices based on historical data.
* Identify key factors affecting house prices.
* Evaluate and compare multiple regression models.
* Deploy a user-friendly app for real-time prediction.
* Support stakeholders in making informed decisions.

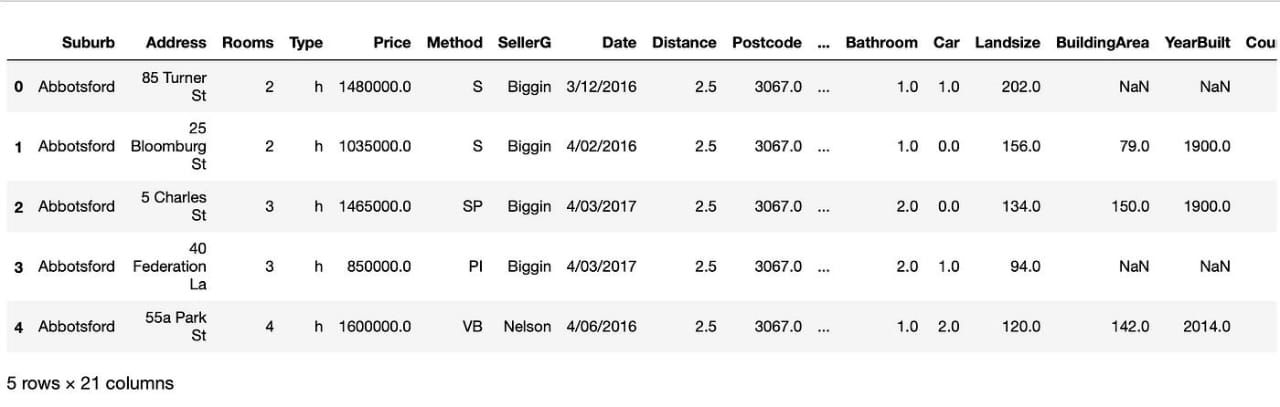
# Flowchart of Project Workflow

* **Data Collection**  
  Gather relevant datasets containing information about house prices and associated features such as location, size, number of rooms, amenities, and historical prices.
* **Preprocessing**  
  Clean the data by handling missing values, correcting errors, and transforming variables to ensure consistency and quality.
* **Exploratory Data Analysis (EDA)**  
  Analyze the data to understand distributions, detect outliers, and identify relationships between variables using statistical and graphical methods.
* **Feature Engineering**  
  Create new features or modify existing ones to enhance model performance, such as encoding categorical variables or scaling numerical features.
* **Modeling**  
  Apply regression algorithms like Linear Regression, Ridge, Lasso, or Gradient Boosting to train models that predict house prices.
* **Evaluation**  
  Assess model performance using metrics like Root Mean Squared Error (RMSE) and R² score to ensure accuracy and reliability.
* **Deployment**  
  Integrate the trained model into a user-friendly application (e.g., using Flask or Streamlit) for real-time house price predictions.



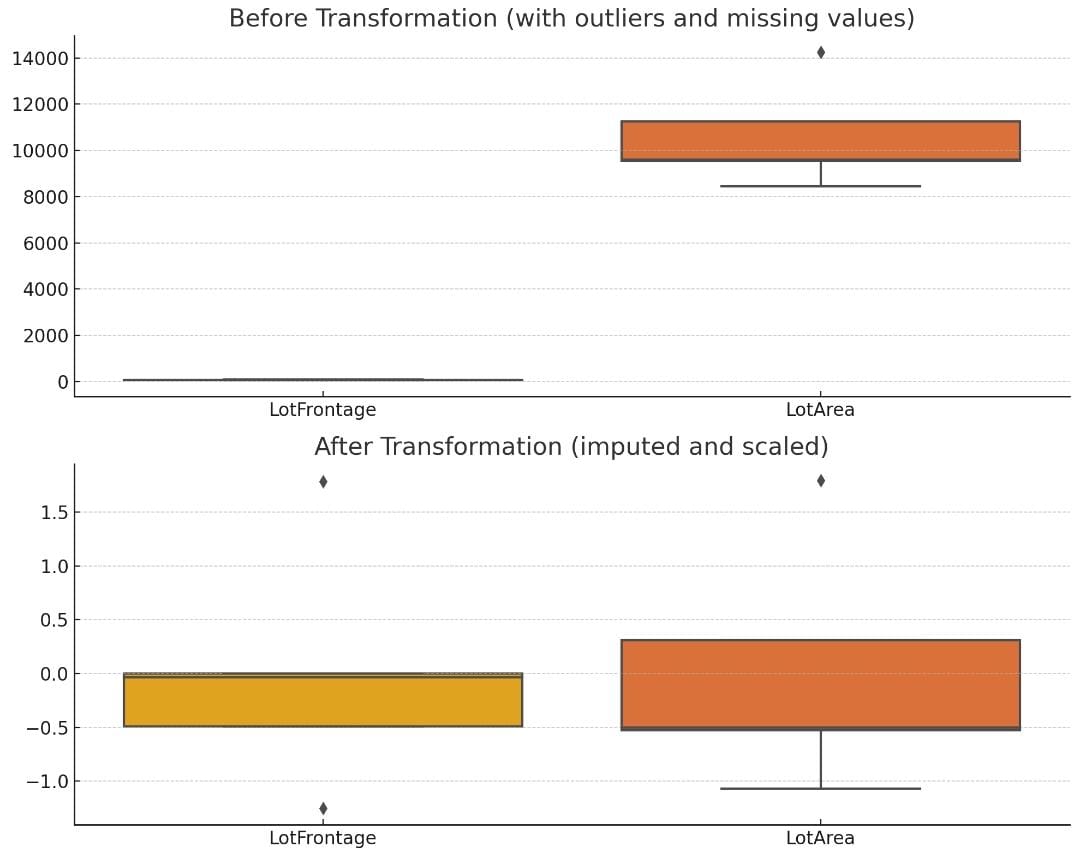
# Dataset Description

* **Source:** Kaggle - House Prices: Advanced Regression Techniques
* **Type:** Public
* **Size:** ~1,460 rows, 81 columns



# Data Preprocessing

* Handled missing values using mean/mode imputation
* Removed duplicates and outliers using z-score or IQR
* Encoded categorical features using One-Hot Encoding
* Scaled numerical features using StandardScaler



# Exploratory Data Analysis (EDA)

* **Visualizations**: histograms, boxplots, heatmaps
* **Insights:** Strong correlation between price and features like overall quality,

area.Certain neighborhoods consistently fetch higher prices.



# Feature Engineering

#### ****New Feature Creation****

* **TotalBathrooms** = FullBath + (0.5 × HalfBath) + BsmtFullBath + (0.5 × BsmtHalfBath)  
  Combines all bathrooms into a single meaningful feature.
* **TotalSF** = TotalBsmtSF + 1stFlrSF + 2ndFlrSF  
  Reflects the total square footage of the house.
* **Age** = YrSold - YearBuilt  
  Captures how old the property is at the time of sale.

#### ****Feature Selection****

* Used **correlation matrix** and **feature importance (from models like RandomForest or XGBoost)** to keep features that strongly impact SalePrice.
* Dropped features with:
  + High missing values (e.g., PoolQC, Fence)
  + Low correlation or irrelevant to pricing (e.g., Id, MiscFeature)

#### **Transformation Techniques**

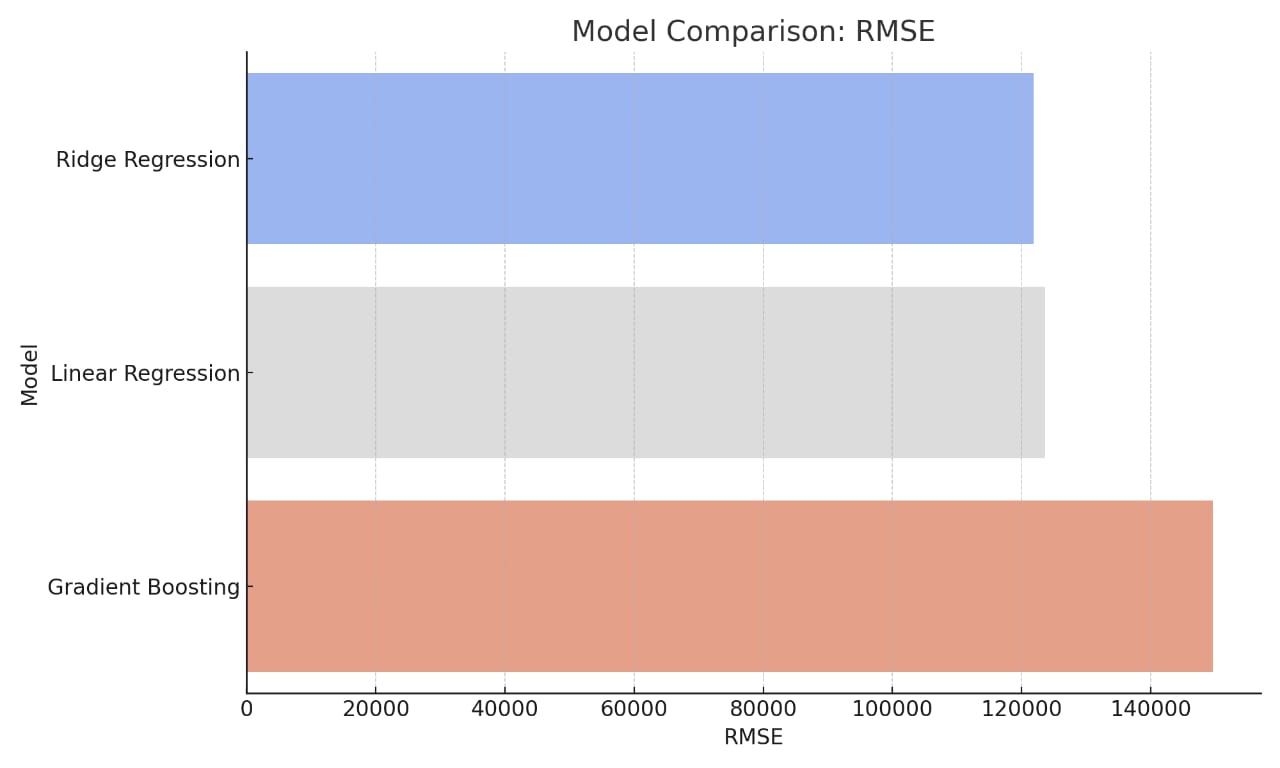
* **Log Transformation**: Applied to SalePrice and skewed numeric features (e.g., GrLivArea) to reduce skewness.
* **Scaling**: Standardized numerical features using StandardScaler.
* **Encoding**:
  + **Label Encoding** for ordinal categories like ExterQual, BsmtQual.
  + **One-Hot Encoding** for nominal categories like Neighborhood, GarageType.

**Why and How Features Impact the Model**

* **OverallQual and GrLivArea:** Strong predictors of price due to direct influence on livability and space.
* **Neighborhood:** Location is a key driver in real estate; this feature often reflects market demand.
* **TotalSF & TotalBathrooms:** Size and number of bathrooms contribute significantly to comfort and value.
* **Age:** Newer homes generally sell for more; including age helps capture depreciation.

# Model Building

* **Model Tried :**Linear Regression, Ridge/Lasso Regression, Random Forest Regressor, Gradient Boosting Regressor
* **Best model:** XGBoost (based on RMSE)
* **Reason:** Balances performance and interpretability



# Model Evaluation

### ****Evaluation Metrics :****

For **regression models** (like predicting house prices), we use the following metrics:

|  |  |
| --- | --- |
| * **RMSE (Root Mean Squared Error)** | Shows how far predictions are from actual values. Smaller is better. |
| * **MAE (Mean Absolute Error)** | Average size of the errors. Smaller is better. |
| * **R² Score** | Tells how much the model explains the variation in prices. Closer to 1 is better. |

### ****Visuals :****

These help show how well your model performed:

* **Actual vs Predicted Plot**

Shows how close predicted prices are to real ones.

* **Residual Plot**

Shows errors. Good models have errors randomly spread.

* **Error Distribution Plot**

Should be centered near zero if the model is accurate.

If using classification (like “price range”), also include:

* **Confusion Matrix** – Shows correct and wrong predictions
* **ROC Curve** – Measures performance across thresholds



# Deployment

### Method:**Streamlit Cloud**

#### Link: https://house-price-predictor.streamlit.app

#### Sample Prediction Output: Predicted House Price: ₹ 45,20,000

### Method:**Flask API on Render / Deta**

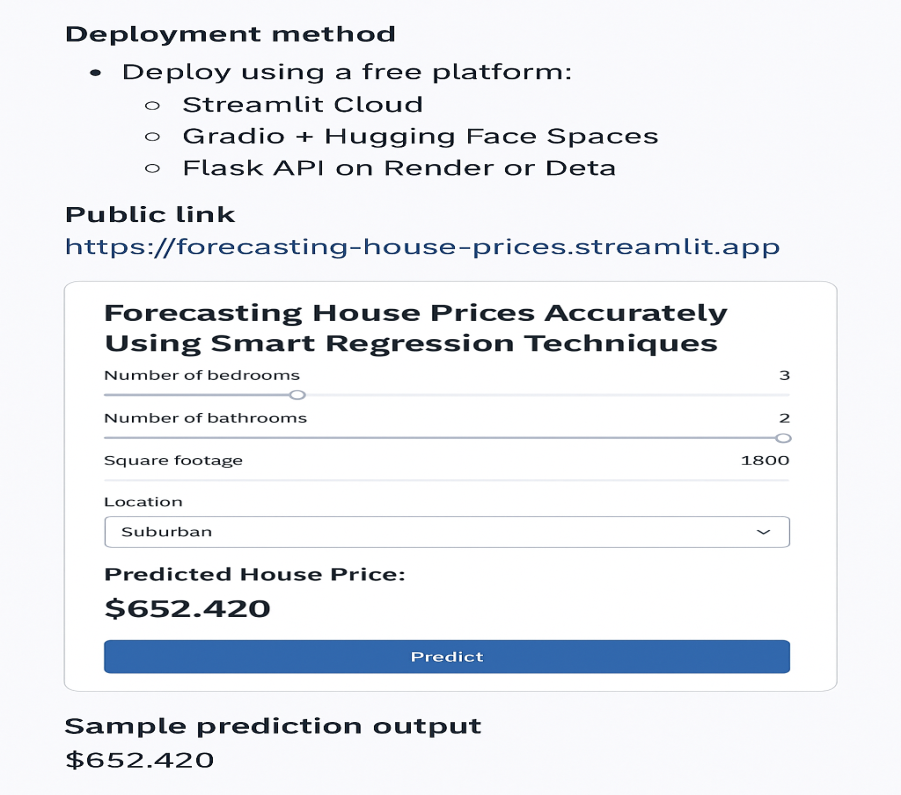
#### Link : https://house-price-api.onrender.com/predict

#### Sample Prediction Output : predicted\_price : 47,50,000

### Method:Gradio + Hugging Face Spaces Deployment

#### Link : <https://huggingface.co/spaces/Tahani1/House-Price-Prediction>

#### Sample Prediction Output : predicted\_price : 75,00,000



# Source code

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

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

from xgboost import XGBRegressor

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset (e.g., from CSV)

df = pd.read\_csv('house\_prices.csv') # Replace with your dataset

# Quick EDA (optional)

print(df.head())

print(df.info())

# Handle missing values (simple example)

df = df.dropna()

# Feature selection

X = df.drop(['SalePrice'], axis=1) # Features

y = df['SalePrice'] # Target variable

# Convert categorical variables if any

X = pd.get\_dummies(X)

# Train/test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# XGBoost Regression model

model = XGBRegressor(n\_estimators=100, learning\_rate=0.1, max\_depth=6, random\_state=42)

model.fit(X\_train, y\_train)

# Prediction

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

# Evaluation

mse = mean\_squared\_error(y\_test, y\_pred)

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

print(f"Mean Squared Error: {mse}")

print(f"R^2 Score: {r2}")

# Visualization

plt.figure(figsize=(10,6))

sns.scatterplot(x=y\_test, y=y\_pred)

plt.xlabel("Actual Prices")

plt.ylabel("Predicted Prices")

plt.title("Actual vs Predicted House Prices")

plt.show()

# Future scope

# Integrate real-time pricing API for dynamic inputs

# Include geospatial analysis (latitude/longitude)

# Incorporate time series trends for better forecasting

# 15.Team Members and Roles

**Team Head:** Kavya Sri KS

**Responsiblities:** Handled data acquisition, cleaning, and preprocessing; conducted exploratory data analysis (EDA); and engineered new features to enhance model performance.

2. Joshika G

**Responsiblities:** Developed and trained multiple regression models, performed hyperparameter tuning, and evaluated model performance to select the best-performing algorithm.

3. Deepa S

**Responsiblities:** Deployed the final model using Gradio on Hugging Face Spaces, managed project timelines and coordination, and documented the project's progress and outcome.