Assignment 2: House■Price Regression with Explainability

Overview: Create a regression model that predicts house prices and explains its predictions. Compare at least two algorithms (e.g., Gradient Boosting vs. Random Forest) and add SHAP■based interpretability.

Objectives: • Perform feature engineering on numerical & categorical attributes. • Train, tune, and compare two regression algorithms. • Provide model interpretability using SHAP or similar. • Package the best model for batch inference.

Dataset: Use the "Ames Housing" dataset from Kaggle (2,930 rows, 80 features).

Tasks: 1. Clean missing data, engineer at least three new features, and justify them. 2. Split data with stratified sampling on price quantiles; perform cross■validation. 3. Tune hyperparameters with GridSearchCV or Optuna. 4. Achieve RMSE ≤■12,000 on the test set. 5. Generate SHAP summary and dependence plots for top 5 features. 6. Save the model (joblib) and a script that loads CSV input and outputs predictions.

Deliverables: • Notebook(s) & scripts. • PDF report (≤■3 pages) with EDA, results, and SHAP visuals. • "run_inference.py" demonstrating batch scoring.

- Clean missing data, engineer at least three new features, and justify them

1. I first removed any columns that had more than 30% missing data. These columns usually don't give us reliable information and can confuse the model if we try to fill them.
2. Filling the remaining missing values**:** For the columns that had fewer missing values, I filled them using Mode for categorical data (the most frequent value Median for numerical data This way, I avoided using mean, which can be affected by outliers
3. I calculated Age of the house by subtracting the year the house was built from the year it was sold. Older houses may lose value over time, while newer houses might be more attractive to buyers Instead of keeping full and half bathrooms separate, for bathrooms I treated a half bathroom as 0.5, since it adds some value but not as much as a full one. This gives a better idea of total usable bathrooms. For the Total square footage area the basement, first floor, and second floor to get the total liveable space in the house. Bigger homes usually sell for more, so this is an important feature

- Split data with stratified sampling on price quantiles, perform cross validation.

transforming the target variable the house sale price using a log scale this helps stabilize the variance and handle the skew in price distribution, which is common in real data

Then, I selected a set of top features that are known to influence housing prices such as size (Gr Liv Area, TotalSF), quality (Overall Qual, Exter Qual, Kitchen Qual), and location (Neighbourhood).

To prepare the data properly, I applied stratified sampling based on price quantiles. This ensures that both training and test sets have a similar distribution of low, medium, and high-priced houses, which makes model evaluation better

I applied 5-fold cross-validation on the training data. This means the training data was split into 5 parts, and the model was trained and validated 5 times, each time on a different fold. This gives us a more reliable and robust estimate of model performance

Cross-validated RMSE (Random Forest): 0.0879

- Tune hyperparameters with GridSearchCV or Optuna.

The training set is split into 5 parts. The model is trained on 4 parts and validated on the 5th, repeated 5 times. This helps avoid overfitting and gives a stable RMSE score.

For each trial, I measured RMSE (Root Mean Squared Error**)** on the original scale of prices (by reversing the log transformation).
Lower RMSE = Better model performance.

- Achieve RMSE £■12,000 on the test set

Train RMSE: ₹9640.47 Test RMSE: ₹11887.78

I evaluated the model on both training and test data using RMSE this shows how accurate the model is in predicting house prices. I first reversed the log transformation to bring values back to the real-world price scale (in ₹), then calculated the average error. This gives us a clear, interpretable metric to understand model performance in good terms.

- Generate SHAP summary and dependence plots for top 5 features.

Generated SHAP summary and dependence plots for top 5 features. Saved the model (joblib) and a script that loads CSV input and outputs predictions.