House Prices Dataset Analysis

Group 4   Kate Chung, Elise Yi Gao, Omair Gill, Yvonne Ye

Wednesday 17th December, 2025

Abstract

We analyze the Ames, Iowa house prices dataset (via Kaggle) and build predictive models for sale price using regression based methods. Our workflow is organized into two stages. First, we conduct exploratory data analysis to understand feature distributions, missingness patterns, and relationships between predictors and sale price. Second, we build and evaluate predictive models, with an emphasis on reproducibility and clear model comparison using a held out test set. Overall, our results show that a relatively small set of structural and quality related features explain a substantial portion of variation in sale price, and that regularized regression can improve generalization when many correlated predictors are present.

Curvenote

1 1. Introduction

Accurate home price prediction is useful for homeowners, buyers, and policymakers because it summarizes how property characteristics translate into market valuation. The Ames housing dataset is a standard benchmark for regression problems because it includes many interpretable predictors and represents common challenges found in real world data, such as skewed variables, missing values, and correlated features.

Our goals are:

1. Describe the dataset and key patterns uncovered during EDA.
2. Build regression models to predict sale price.
3. Compare models on a held out test set using consistent evaluation metrics.
4. Package our work so it can be run end to end (environment, functions, tests, and notebooks).

2 2. Data

We use the publicly available "House Prices Dataset" from Kaggle: https://www.kaggle.com/datasets/lespin/hc
prices-dataset

The dataset contains observations of houses in Ames, Iowa, with a response variable (sale price) and a set of predictors describing size, quality, neighborhood, and other attributes. The version used for this project is already cleaned and split into training and testing sets, which supports a straightforward modeling pipeline.

In our repository, the analysis is organized as:

- Part 1 (EDA): `Part1-EDA/EDA.ipynb`
- Part 2 (Prediction): `Part2-Prediction/classification.ipynb`

3 3. Exploratory Data Analysis

3.1 3.1 Distribution of sale prices

We begin by examining the distribution of sale price. Housing prices are typically right skewed, and in many regression settings a log transform helps stabilize variance and improve linear model fit. We compare modeling on raw sale price versus log transformed sale price and carry forward the representation that yields better calibration and residual behavior.

3.2 3.2 Missingness and data quality checks

We check for missingness across predictors, identify variables with substantial missing values, and decide on a consistent strategy for handling them. Depending on the variable type, we use either:

- simple imputation for numeric variables (for example median imputation), or
- explicit "missing" categories for categorical predictors, when missingness is meaningful.

We also verify basic data integrity (ranges, impossible values, duplicates where relevant).

3.3 3.3 Feature relationships and collinearity

We explore correlations among numeric predictors and their relationship with sale price. Common high impact predictors include living area, overall quality, and garage capacity, while neighborhood related predictors often capture location effects. We also look for multicollinearity among size related variables; this informs whether regularization is likely to help.

3.4 3.4 Key takeaways from EDA

From EDA, we identify a small set of predictors that appear strongly associated with sale price. These relationships guide our baseline model and feature engineering choices. We also flag any outliers that may influence least squares fit and decide whether to keep or robustly handle them.

4 4. Modeling Approach

4.1 4.1 Train/test setup and evaluation metrics

We train models on the provided training split and evaluate on the held out test split. For model comparison, we use consistent metrics such as:

- RMSE (root mean squared error),
- MAE (mean absolute error), and
- (R2) on the test set.

If we train on log sale price, we evaluate both in log space and in original price units (after back transform), noting that back transforming introduces bias unless corrected.

4.2 4.2 Baseline model

We begin with a baseline linear regression using a core set of predictors chosen for interpretability and coverage of major price drivers. This baseline provides a reference point for more complex models and helps reveal underfitting.

4.3 4.3 Feature engineering and preprocessing

We apply preprocessing steps consistently across models:

- separate pipelines for numeric and categorical features,
- scaling for numeric features when needed (especially for regularized regression),
- one hot encoding for categorical variables,
- imputation for missing values.

We implement reusable preprocessing and modeling functions in `src/` and validate them with unit tests in `tests/`.

4.4 4.4 Regularized regression (if used)

To address multicollinearity and reduce variance, we consider regularized regression:

- Ridge regression (L2 penalty),
- Lasso regression (L1 penalty), or
- Elastic Net (combined penalties).

We select hyperparameters using cross validation on the training set, then evaluate the final tuned model on the test set.

4.5 4.5 Model selection criteria

We select a final model based on test performance and stability. When performance differences are small, we prioritize interpretability and robustness. We also examine residual plots and error patterns to understand when the model performs poorly (for example, very expensive homes, unusual property types, or sparse categories).

5 5. Results

5.1 5.1 Predictive performance

Overall, the baseline linear regression performs reasonably, but regularization and a stronger preprocessing pipeline can improve test performance, particularly when the feature set includes many correlated predictors and one hot encoded categories.

5.2 5.2 Interpretability

We inspect fitted coefficients (or feature importances for the models used) to identify key drivers of predicted sale price. Common themes include:

- quality and condition variables,
- house size and usable living area,
- garage and basement features,
- neighborhood or location proxies.

We interpret these results cautiously, since coefficients reflect association rather than causal effects, and correlated predictors can shift coefficient magnitudes.

5.3 5.3 Error analysis

We analyze where the model makes larger errors. Common patterns include:

- higher variance in predictions for expensive homes,
- systematic underprediction or overprediction in certain neighborhoods,
- sensitivity to outliers.

We include residual plots and predicted versus actual plots to visualize fit.

6 6. Discussion

This project demonstrates how a structured workflow, starting from careful EDA and proceeding to reproducible modeling, can produce strong predictive performance on a classic regression dataset. Our results support the idea that a relatively small set of high-level features explain much of the variation in housing prices, while regularization and consistent preprocessing help stabilize performance when the model includes many predictors.

At the same time, this analysis is predictive rather than causal. The dataset is specific to Ames, Iowa and a particular time period, so model performance and feature effects may not generalize to other markets without retraining and validation.

7 7. Limitations and Future Work

Limitations include:

- limited external validity outside Ames, Iowa,
- potential bias introduced by preprocessing choices (especially for rare categories),
- performance sensitivity at the high end of the price distribution.

Future work could include:

- trying nonlinear models (random forests, gradient boosting) for potentially better accuracy,
- adding interaction terms for key predictors,
- using more robust methods for outliers,
- calibration checks after back transforming from log scale (if used).

8 8. Reproducibility

This repository is designed to be reproducible:

- Environment: `environment.yml`
- Functions: `src/`
- Tests: `tests/`
- Notebooks: `Part1-EDA/` and `Part2-Prediction/`

To run tests:

```
pytest tests/
```