House Prices Prediction

Kaggle: Advanced Regression Techniques

Abstract

This report describes a machine learning project built on the dataset provided by the *House Prices — Advanced Regression Techniques* competition hosted on Kaggle. The objective is to develop models that accurately predict residential house sale prices using a rich set of explanatory variables. The dataset consists of detailed characteristics of homes in Ames, Iowa, and includes both numeric and categorical variables. This document summarizes the data, preprocessing steps, modeling approaches, evaluation metrics, and key results.

1 Introduction

The objective of this project is to predict the final sale price of homes using the *House Prices — Advanced Regression Techniques* dataset from Kaggle. This is a supervised regression task where the target variable is `SalePrice`, and explanatory variables describe various aspects of the residential properties. The competition provides both training and test sets, with 79 feature columns and one target in the training set. :contentReference[oaicite:1]index=1

2 Dataset Description

The dataset used in this competition consists of two primary files: `train.csv` and `test.csv`. The training set contains 1460 instances with 80 columns, while the test set contains 1459 instances with 79 columns (excluding the target). Each row describes a particular home, and the columns include a mixture of numerical, categorical, and ordinal features. :contentReference[oaicite:2]index=2

2.1 Target Variable

The target variable is:

- **SalePrice** — the selling price of the house (in USD). This is the variable that the models attempt to predict. :contentReference[oaicite:3]index=3

2.2 Feature Variables

The dataset includes 79 explanatory variables representing various attributes of homes. These features describe physical characteristics, quality/condition ratings, location information, and more. Common examples include:

- `OverallQual` — overall material and finish quality.
- `GrLivArea` — above ground living area (square feet).
- `GarageCars` — size of garage in car capacity.
- `TotalBsmtSF` — total basement area.
- `YearBuilt` — year of construction. :contentReference[oaicite:4]index=4

These features cover a wide range of residential house characteristics, enabling models to learn relationships between property attributes and sale prices.

2.3 Data Statistics

The typical structure of the data can be summarized as:

- **Training instances**: 1460
- **Test instances**: 1459
- **Total features**: 79
- **Target variable (train only)**: `SalePrice` :contentReference[oaicite:5]index=5

The features span both numerical and categorical types, requiring careful preprocessing and encoding.

3 Data Preprocessing

Preprocessing typically includes:

- Handling missing values via imputation based on semantic meaning (e.g., `None` for missing basements).
- Encoding categorical features using techniques such as One-Hot Encoding or Ordinal Encoding for ordered quality categories.
- Scaling/normalizing numerical features where appropriate (e.g., for linear/SVR models).
- Train–validation split to ensure the model is evaluated on unseen data.

4 Modeling Approaches

Various models were trained and evaluated, including:

- Linear Regression
- Ridge and Lasso Regression
- Decision Tree Regressor
- Random Forest Regressor
- Gradient Boosting Regressor
- Support Vector Regressor
- XGBoost Regressor

Each model was trained using appropriate preprocessing and evaluated using regression metrics such as R^2, RMSE, MAE, and MAPE.

5 Evaluation Metrics

Common regression metrics used include:

- **R^2** — coefficient of determination, measuring explained variance.
- **RMSE** — root mean squared error, sensitive to large errors.
- **MAE** — mean absolute error.
- **MAPE** — mean absolute percentage error.

6 Key Results

A subset of results is shown in Table 1. These values summarize model performance on training and test datasets.

Model	**R^2 (Test)**	**RMSE**	**MAE**	**MAPE**
Linear Regression	0.874	31103	20108	0.123
Ridge Regression	0.879	30485	19380	0.116
Random Forest	0.893	28601	17346	0.105
Gradient Boosting	0.897	28102	17296	0.102
XGBoost (Tuned)	0.908	26557	16909	0.103

Table 1: Regression model performance summary

These results demonstrate that tree-based ensemble methods, especially XGBoost, yield superior performance on the Ames housing dataset for this task.

7 Residual Analysis

Residual analysis showed that the model predictions were generally unbiased with normally distributed errors in the central range. Some heteroscedasticity was observed at higher sale prices, which is a common characteristic of housing data.

8 Prediction Intervals

Prediction intervals were estimated using bootstrap methods, yielding an empirical coverage of approximately 81% for nominal 95% intervals. This undercoverage suggests that the intervals underestimated uncertainty and could be improved via methods such as quantile regression.

9 Future Deployment

This project can be extended into a production-ready system capable of accepting raw input data and returning price predictions automatically. The trained preprocessing and modeling pipeline is designed to handle raw input in the same format as the original dataset, making deployment straightforward.

9.1 Input Handling

The deployed system will accept a raw CSV file containing house attributes with the same schema as the training dataset (excluding the target variable `SalePrice`). This file may contain missing values and categorical variables in their original string format. The system will:

- Validate the input schema and ensure required columns are present.
- Apply the same preprocessing steps used during training, including:
  - Missing value imputation,
  - Ordinal encoding for quality-related features,
  - One-hot encoding for nominal categorical variables,
  - Numerical feature scaling where applicable.

Because preprocessing is embedded inside a unified pipeline, no manual feature engineering is required at inference time.

9.2 Prediction Pipeline

Once preprocessing is complete, the processed data is passed directly into the trained regression model (XGBoost). The model outputs a predicted sale price for each row in the input file. These predictions are then appended as a new column (e.g., `PredictedSalePrice`) to the original dataset.

9.3 System Architecture

The deployment workflow can be summarized as:

1. User uploads a raw CSV file.
2. The system loads the trained pipeline from disk.
3. The pipeline preprocesses the input features.
4. The trained model generates predictions.
5. A new CSV file with predictions is returned to the user.

This process can be implemented as:

- A command-line script for batch prediction,
- A REST API using frameworks such as FastAPI or Flask,
- Or a web-based interface built with Streamlit.

9.4 Benefits

This deployment strategy ensures:

- Consistency between training and inference preprocessing,
- Minimal manual intervention by the user,
- Scalability to large datasets,
- Reproducibility and robustness of predictions.

Overall, this allows the model to function as a reliable decision-support tool capable of generating house price predictions from raw real-world data with minimal effort.

10 Conclusion

This project demonstrates an end-to-end approach to predicting house prices using a rich tabular dataset from Kaggle. Effective preprocessing and model selection were critical to achieving strong predictive performance. XGBoost with tuned hyperparameters provided the best results, and residual diagnostics indicated sound model behavior. Further work could explore deeper uncertainty quantification and richer feature engineering.

11 References

- **House Prices — Advanced Regression Techniques**, Kaggle competition page and associated data description. :contentReference[oaicite:6]index=6