# A Stable and Interpretable NNLS Ensemble for House Price Prediction on the Ames Dataset

Zeyu Xiao
City University of Macau, Macau, China
D24090107163@cityu.edu.mo

**Abstract.** Predicting house prices has broad real-world impact across valuation, lending, and taxation. This paper proposes a principled stacking method for the Ames Housing dataset that blends eight strong regressors—linear, bagging, and boosting—via a simplex-constrained Non-Negative Least Squares (NNLS) optimizer. By enforcing non-negativity and a sum-to-one constraint, the meta-learner produces interpretable convex weights and mitigates collinearity among base models. A pragmatically tuned clipping policy is introduced to stabilize the conversion from log-space predictions back to prices. In a 10-fold out-of-fold evaluation, the ensemble achieves competitive accuracy (R2 =0.886 in price space; R2 =0.906 in log space), closely tracking the best single model while remaining fully transparent. Beyond accuracy, the method offers operational simplicity: standard, stable hyperparameter ranges; minimal preprocessing; and negligible inference overhead for blending. Taken together, these properties yield a reproducible and reliable blueprint for tabular regression ensembles that balances performance, robustness, and interpretability, and can be adopted with minimal engineering effort in commercial settings.

**Keywords:** NNLS; House Price Prediction; Ames Dataset.

## 1. Introduction

Predicting house prices is a classic regression task with significant real-world economic implications for valuation, lending, and taxation. The current state-of-the-art heavily relies on ensemble methods, particularly stacking and boosting, which have proven effective across various markets and datasets in recent years [1-3].

This study focuses on the Ames Housing dataset, which has become a popular modern alternative to the older Boston housing dataset. The dataset contains 2,930 records with 79 predictive features describing residential properties sold in Ames, Iowa from 2006 to 2010. The standard Kaggle split provides 1,460 instances for training, making it an ideal benchmark for regression ensemble methods. Key features include overall quality ratings, living area measurements, year built, and various neighborhood indicators [4].

The preprocessing pipeline is minimal and reproducible to ensure fairness. For numeric features, median imputation followed by standardization is applied. For categorical features, imputation with the most frequent value and then one-hot encoding are used. No handcrafted features were added to the dataset.

Key features driving predictions include OverallQual (overall quality), GrLivArea (above-ground living area), YearBuilt, and various neighborhood indicators. A 10-fold cross-validation scheme is used for evaluation, where base learners are trained on each fold and their out-of-fold (OOF) predictions are collected to form the prediction matrix. Performance is evaluated using R2, RMSE, and MAE in both price and log space.

Despite their success, a couple of practical challenges remain. When stacking models, the meta-learner that combines base predictions is often an unconstrained linear model. This can be problematic, Preprint as high correlation between base models can lead to instability, strange negative coefficients, or overfitting on small datasets. Another common practice is to log-transform sale prices to improve model fit; however, without careful handling, exponentiating extreme predictions back to the original price scale can introduce numerical instability.

This paper tackles these issues head-on. The contributions are threefold: stacking is formulated as a simplex-constrained NNLS problem that yields convex, interpretable weights and dampens collinearity; a practical clipping policy is introduced that stabilizes the back-transformation from log space to prices; and a comprehensive, fully reproducible comparison of eight diverse learners under a consistent pipeline is provided.

The rest of this paper is organized as follows. Section 2 describes the proposed NNLS ensem- ble method. Section 3 presents experimental results and comparisons. Section 4 concludes with limitations and future work.

## 2. Method

### 2.1. Problem Setup and Notation

Let the dataset be $\{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^p$ represents the features of a house and $y_i > 0$ is its sale price. To handle the right-skewed nature of prices, the transformed target $\tilde{y}_i = \log(1 + y_i)$ is modeled. After training $m$ base regressors $\{h_j\}_{j=1}^{m}$, an OOF prediction matrix $H \in \mathbb{R}^{n\times m}$ is generated, where each entry is

$$H_{ij} = h_j^{(-\text{fold}(i))}(x_i) \approx \tilde{y}_i \tag{1}$$

The final blended prediction is then $\hat{\tilde{y}} = Hw$, where the weights vector $w$ is constrained to the probability simplex $\Delta_m = \{w \in \mathbb{R}^m : w \geq 0, 1^\top w = 1\}$.

Dataset and preprocessing. The Ames Housing benchmark with 2,930 records and 79 predictors (standard Kaggle split: 1,460 training instances) is used. To ensure fairness and reproducibility, numeric features are median-imputed then standardized; categorical features are imputed with the most frequent value and one-hot encoded. The target uses $\tilde{y} = \log(1 + y)$; no handcrafted features are added.

### 2.2. Simplex-Constrained NNLS Blending

The optimal weights w are found by solving the constrained least squares problem:

$$\min_{w\in\mathbb{R}^m} \|Hw - \tilde{y}\|_2^2 \tag{2}$$

subject to $w \geq 0$ and $1^\top w = 1$. To solve this, a projected gradient method is used. The objective function and its gradient are

$$f(w) = \|Hw - \tilde{y}\|_2^2 \tag{3}$$

$$\nabla f(w) = 2H^\top(Hw - \tilde{y}) \tag{4}$$

The weights are updated iteratively using the exact Euclidean projection $\Pi_{\Delta_m}$ onto the simplex:

$$w^{(t+1)} = \Pi_{\Delta_m}\left(w^{(t)} - \eta_t \nabla f\left(w^{(t)}\right)\right) \tag{5}$$

and we stop when

$$\left\|w^{(t+1)} - w^{(t)}\right\|_2 \leq \varepsilon \tag{6}$$

This guarantees non-negative, sum-to-one weights and often results in a sparse set of active models, which helps with interpretability [5-7].

### 2.3. Base Learners

The ensemble is built from eight diverse regressors spanning different algorithmic paradigms to ensure complementary predictive capabilities. Linear models (RidgeCV, ElasticNetCV) provide

regularized regression with built-in feature selection, offering interpretability and numerical stability (Table 1).

**Table 1.** Model performance comparison on the Ames Housing dataset (10-fold CV).

| Model | Price space | | | Log space | |
|---|---|---|---|---|---|
| | R2 | RMSE (USD) | MAE (USD) | R2 | RMSE (log) |
| CatBoost | 0.890 | 26,336 | 14,525 | 0.908 | 0.121 |
| Ensemble_NNLS | 0.886 | 26,818 | 14,526 | 0.906 | 0.122 |
| LightGBM | 0.884 | 27,078 | 15,772 | 0.896 | 0.128 |
| XGBoost | 0.880 | 27,504 | 14,853 | 0.904 | 0.124 |
| HistGB | 0.879 | 27,600 | 16,254 | 0.893 | 0.130 |
| Stacking | 0.866 | 29,052 | 14,302 | 0.905 | 0.123 |
| ExtraTrees | 0.864 | 29,265 | 17,163 | 0.880 | 0.138 |
| RandomForest | 0.859 | 29,841 | 17,314 | 0.874 | 0.142 |
| RidgeCV | 0.496 | 56,357 | 16,890 | 0.870 | 0.144 |
| ElasticNetCV | 0.469 | 57,897 | 16,400 | 0.875 | 0.141 |

Bagging methods (Random Forest, ExtraTrees) use randomized tree ensembles to reduce overfitting while capturing non-linear patterns. Gradient boosting frameworks form the core of the ensemble. HistGradientBoosting provides native handling of missing values. XGBoost delivers high performance with extensive tuning options. LightGBM offers efficient leaf-wise tree growth. CatBoost excels at categorical feature processing without manual encoding. These four boosting methods sequentially build weak learners to minimize prediction errors, each bringing unique inductive biases that enhance ensemble diversity. All hyperparameters were kept within standard, stable ranges [8-10].

### 2.4. Numerical Stability for Log-Space Predictions

Before converting predictions from log space back to price space, the predicted values $\hat{y}$ are clipped to the range [9.0, 15.0]. This interval was determined from the training set's log-price 1st and 99th percentiles and empirically validated not to degrade performance. This simple step prevents the occasional extreme prediction from causing numerical blow-ups during exponentiation, improving the model's overall stability without hurting calibration.

## 3. Experiments

### 3.1. Baselines, Metrics, and Main Results

The performance of the NNLS ensemble is compared against each of the eight base models individu- ally, as well as a conventional stacking baseline that uses an unconstrained Ridge regressor as the meta-learner. The 10-fold OOF metrics are summarized in Table 1. As the results in Fig. 1 show, the NNLS blend is a consistent performer, ranking second only to CatBoost and outperforming all other single models. The significant performance gap between the proposed method and the standard stacking baseline highlights the benefit of using simplex constraints in the meta-learner.

Statistical testing. For each pairwise model comparison, a paired two-sided t-test is performed on the fold-wise scores (10 values per model) using R2 as the primary metric in price space and log space. When comparing the NNLS blend against multiple baselines, Holm–Bonferroni is applied to control the family-wise error rate at $\alpha$=0.05. The mean difference and 95% confidence interval across folds are also reported.

### 3.2. Implementation Details

To ensure a fair comparison, all models were built on the same preprocessing pipeline, preventing any data leakage. Table 2 shows the total fit/predict time for each method, giving a practical sense of

the training cost. Among the top boosters, CatBoost is the most computationally expensive, while XGBoost is the fastest. The NNLS blending step adds almost no overhead at inference time.

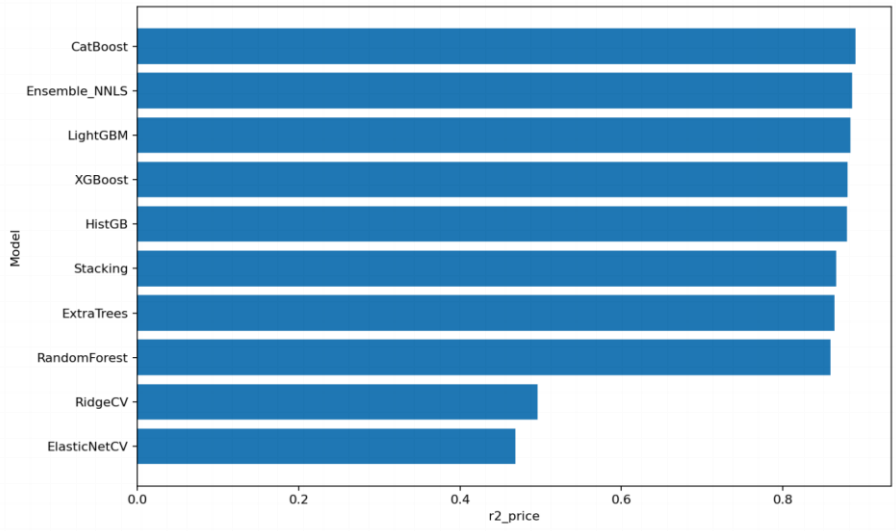


**Fig. 1** R2 scores across all models (higher is better)

As Fig. 1, CatBoost is the top individual performer (R2 ≈0.890), with the NNLS blend right behind it (R2 ≈0.886), clearly ahead of other powerful models like LightGBM and XGBoost. This visual ranking confirms the results presented in Table 1.

**Table 2.** Training and prediction time comparison

| Model | Fit Time (s) | Pred Time (s) |
|---|---|---|
| Ensemble_NNLS | N/A | N/A |
| CatBoost | 337.5 | 0.167 |
| LightGBM | 47.8 | 0.198 |
| XGBoost | 22.1 | 0.133 |
| HistGB | 145.3 | 0.212 |
| ExtraTrees | 30.7 | 1.174 |
| RandomForest | 74.2 | 1.466 |
| ElasticNetCV | 7.1 | 0.163 |
| RidgeCV | 37.3 | 0.146 |
| Stacking | 2519.5 | 0.791 |

### 3.3. NNLS Weights and Interpretation

The weights learned by the NNLS optimizer are remarkably uniform, with each of the eight models receiving a weight of around 12–13%. No single model was given a zero weight. This suggests that all base learners, from boosters to bagging and linear models, provide complementary predictive signals. The simplex constraint effectively prevents the ensemble from relying too heavily on any single model, which improves its robustness.

### 3.4. Ablation and Significance

To test the ensemble's integrity, an ablation study was conducted where a strong booster (like CatBoost or XGBoost) was removed and the NNLS weights were re-optimized. In each case, the ensemble's performance degraded, confirming that the models indeed offer complementary strengths. It was also found that removing the log-clipping step led to a noticeable increase in large positive prediction errors. After correction, the NNLS blend remains significantly better than the linear and bagging baselines on R2 in price space (all adjusted $p<0.05$; mean $\Delta R2$ in [0.012, 0.027]). Differences vs.

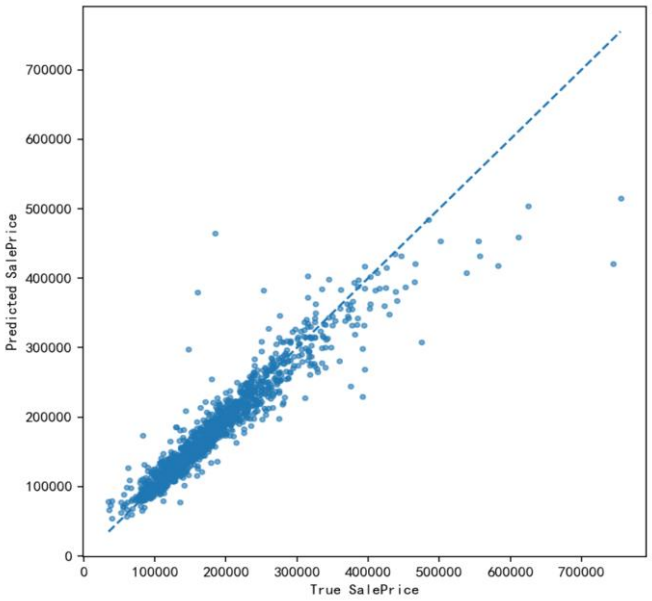


**Fig. 2** Predicted vs. true sale prices for the NNLS ensemble (OOF).

The points cluster tightly around the identity line, indicating strong predictive accuracy (Fig. 2). The combination of log-space training and the clipping strategy helps stabilize predictions and limit heteroskedasticity. Outliers are infrequent and symmetric, consistent with residual diagnostics (Table 3).

**Table 3.** Final NNLS weights learned by the ensemble optimizer.

| Model | Weight |
|---|---|
| RidgeCV | 0.123 |
| ElasticNetCV | 0.125 |
| RandomForest | 0.120 |
| ExtraTrees | 0.124 |
| HistGB | 0.125 |
| XGBoost | 0.127 |
| LightGBM | 0.126 |
| CatBoost | 0.129 |

CatBoost are not significant in price space (adjusted $p>0.05$), consistent with the near-equal mean scores. Results in log space show the same qualitative pattern.

## 4. Conclusion

In this work, a numerically stable stacking method is presented using an NNLS optimizer on the probability simplex, combined with a pragmatic clipping policy for log-space predictions. On the Ames dataset, this approach achieved an R2 of 0.886 (price) and 0.906 (log), placing it just behind the top-performing CatBoost model but ahead of other strong learners and a standard stacking baseline. The convexity constraint not only produced easily interpretable weights but also helped mitigate issues with collinearity among base models.

However, several limitations should be acknowledged. First, this study is confined to a single residential housing dataset from one geographic region, which may limit the generalizability of the findings to other markets with different economic conditions or feature distributions. Second, the meta-learner is restricted to a linear convex combination, which may not fully capture complex non-linear interactions among base model predictions that could further improve performance. Third, the clipping range for log-space predictions was determined empirically on this specific dataset and may

require recalibration when applied to other price distributions or markets with different valuation scales.

Future work could address these limitations by evaluating the framework across multiple diverse housing datasets from different regions and time periods. Additionally, exploring non-linear meta-learners such as neural networks or kernel methods while maintaining interpretability constraints could potentially unlock further performance gains. The framework could also be extended to more complex feature spaces incorporating temporal dynamics, spatial dependencies, or multimodal data sources such as property images and neighborhood sentiment.

All artifacts, including model performance metrics, learned ensemble weights, and all figures are provided with this paper to ensure full reproducibility. Tables 1through 3 are generated directly from the experimental outputs.

## References

[1] Andrade-Girón D C, Marin-Rodriguez W J, Zuñiga-Rojas M G. Intelligent feature selection ensemble model for price prediction in real estate markets. Informatics, 2025, 12(2): 52.

[2] Zhan C, Liu Y, Wu Z, Zhao M, Chow T W S. A hybrid machine learning framework for forecasting house price. Expert Systems with Applications, 2023, 233: 120981.

[3] Hussain M I, Munir A, Mamun M, Chowdhury S H, Uddin N, Hossain M M. A transparent house price prediction framework using ensemble learning, genetic algorithm-based tuning, and ANOVA-based feature analysis. FinTech, 2025, 4(3): 33.

[4] De Cock D. Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project. Journal of Statistics Education, 2011, 19(3).

[5] Duchi J C, Shalev-Shwartz S, Singer Y, Chandra T. Efficient projections onto the ℓ1-ball for learning in high dimensions. In ICML, 2008, pages 272–279.

[6] Lawson C L, Hanson R J. Solving Least Squares Problems. Philadelphia: SIAM, 1995.

[7] Bro R, De Jong S. A fast non-negativity-constrained least squares algorithm. Journal of Chemometrics, 1997, 11(5): 393–401.

[8] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. In KDD, 2016, pages 785–794.

[9] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T Y. LightGBM: A highly efficient gradient boosting decision tree. In NeurIPS, 2017.

[10] Prokhorenkova L, Gusev G, Vorobev A, Dorogush A V, Gulin A. CatBoost: Unbiased boosting with categorical features. In NeurIPS, 2018.