Predicting Housing Prices Using Classical and Literature-Based Machine Learning Models

Shravani Sajekar, University of North Carolina at Charlotte

Abstract— This project investigates the problem of housing price prediction using a combination of classical machine learning models and modern literature-based ensemble methods. The goal is to evaluate how different modeling approaches perform on the Kaggle House Prices dataset and to reproduce methods proposed in two recent research papers. The workflow includes comprehensive preprocessing, feature engineering, implementation of baseline models such as Linear Regression, KNN, and Neural Networks, and reproduction of two state-of-the-art approaches based on XGBoost and Bayesian-optimized ensemble models. Model performance is assessed using metrics including RMSE, MSE, MAE, and R^2, and visual analyses are conducted to understand prediction behavior and residual patterns. The results demonstrate that literature-based ensemble models significantly outperform classical models, highlighting the importance of hyperparameter tuning and advanced boosting techniques in structured data regression tasks.

Index Terms— House price prediction, machine learning, XGBoost, Bayesian optimization, regression models, ensemble learning.

INTRODUCTION

Predicting housing prices is a long-standing problem in machine learning and has become increasingly relevant with the growth of large, publicly available real estate datasets. Accurate price estimation supports buyers, sellers, financial institutions, and policy researchers by offering insight into market behavior and property valuation. However, the task remains challenging due to the heterogeneous nature of housing attributes, complex nonlinear relationships, and the presence of both numerical and categorical variables requiring substantial preprocessing.

This project investigates the housing price prediction problem using the "*House Prices: Advanced Regression Techniques"* dataset from Kaggle, a benchmark dataset widely used in academic and industrial regression tasks. The primary objective is to evaluate the performance of classical machine learning models taught in the course, Linear Regression, K-Nearest Neighbors, and Neural Networks against state-of-the-art approaches proposed in two recent research papers. To accomplish this, the project reproduces the preprocessing steps, modeling strategies, and tuning procedures described in the literature, enabling a direct comparison between widely used baseline methods and more advanced, ensemble-based algorithms.

The primary objectives of this work are:

1. To design a rigorous preprocessing and feature-engineering pipeline suitable for high-dimensional structured data.
2. To build and evaluate classical regression models as foundational baselines.
3. To reproduce and critically examine two research papers that apply advanced machine learning techniques to similar regression tasks.
4. To provide a comprehensive comparison of all models through consistent evaluation metrics, visual diagnostics, and interpretive analysis.

The study follows a unified pipeline beginning with data cleaning, imputations, feature engineering, and transformation of categorical variables using sparse matrix techniques to ensure scalability.

Classical models are implemented using scikit-learn, while literature-based methods incorporate XGBoost and Bayesian-optimized ensemble techniques. Each model is trained on the same processed dataset and evaluated using industry-standard metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and coefficient of determination (R^2). Evaluation also includes visual analysis through residual plots and actual-versus-predicted comparisons to better understand the behavior of each model.

The overarching goal of this work is to determine whether the advanced methods proposed in the literature offer measurable improvements over classical machine learning models and to analyze the factors contributing to differences in model performance. Through this comparison, the project highlights the strengths and limitations of each family of methods and provides insight into best practices for structured tabular regression problems.

METHODOLOGY

A. Dataset Overview

This study uses the *House Prices: Advanced Regression Techniques* dataset available on Kaggle, which contains 79 explanatory variables describing various physical, structural, and environmental characteristics of residential properties in Ames, Iowa. The dataset includes both numerical and categorical attributes, covering aspects such as lot size, building materials, neighborhood classification, interior quality, and house age. The target variable is *SalePrice*, a continuous variable representing the property's final sale amount. The dataset also contains missing values, non-standard categorical encodings, and mixed data types, necessitating a robust preprocessing pipeline before modeling.

B. Preprocessing and Feature Engineering

A unified preprocessing pipeline was developed to ensure that all models were trained using identical representations of the data.Missing values in numerical features were imputed using median values, while categorical variables were imputed using mode. Highly skewed numerical features were transformed using logarithmic or Box–Cox transformations to stabilize variance, particularly for variables known to impact sale price nonlinearly. Categorical variables were encoded using one-hot encoding, resulting in a high-dimensional sparse feature matrix suitable for both tree-based and linear models. The final dataset was partitioned into training and validation sets to enable objective evaluation across all models. The same processed features were shared between classical and literature-based approaches to maintain fairness in comparison.

1) Handling Missing Values

Missing values were addressed using statistical imputation:

- Numerical features x_{num} were imputed using the median:

$$x_{num}^{(i)} = \begin{cases} x_{num}^{(i)}, & \text{if not missing} \\ \text{median}(x_{num}), & \text{otherwise} \end{cases}$$

- Categorical features x_{cat} were imputed using the mode:

$$x_{cat}^{(i)} = \begin{cases} x_{cat}^{(i)}, & \text{if not missing} \\ \text{mode}(x_{cat}), & \text{otherwise} \end{cases}$$

This ensured that no row was discarded, preserving the full dataset structure.

2) One-Hot Encoding for Categorical Variables

Categorical variables were transformed via one-hot encoding, generating a sparse high-dimensional representation. Numerical features were standardized:

$$x' = \frac{x - \mu}{\sigma}$$

This expansion enabled classical linear models and neural networks to incorporate categorical interactions explicitly. The resulting dataset, after encoding, contained several hundred binary indicator variables. Storing these in a sparse matrix format significantly reduced memory usage and improved training efficiency across all models. Scaling helps neural networks and distance-based models converge more effectively.

3) Feature Scaling

Numerical variables were standardized using the StandardScaler. Because the dataset is stored as a sparse matrix, scaling was performed with "with_mean=False" to preserve sparsity. Scaling ensured that distance-based models (e.g., KNN) and gradient-based optimizers (e.g., neural networks) performed reliably.

4) Train-Validation Split

The processed data was split into training and validation sets in an 80/20 ratio:

$$(X_{\text{train}}, y_{\text{train}}), (X_{\text{val}}, y_{\text{val}})$$

This enabled unbiased evaluation of generalization performance.

C. Classical Machine Learning Models

Three baseline models commonly taught in introductory machine learning were implemented using scikit-learn. The Linear Regression model served as the fundamental benchmark, assuming a linear relationship between features and sale price. Despite its simplicity, its interpretability and efficiency make it a useful reference point. The K-Nearest Neighbors Regressor was implemented to assess how well instance-based learning performs on a heterogeneous feature space; the model's sensitivity to feature scaling and local structure provided insight into the need for more complex representations. The Neural Network model used a multilayer perceptron architecture with ReLU activation and adaptive learning rate optimization. This model captured nonlinear relationships but required careful tuning to avoid overfitting, given the dataset's relatively modest size.

1) Linear Regression

Linear Regression models the relationship between features and target using a linear combination:

$$\hat{y} = \beta_0 + \sum_{i=1}^{d} \beta_i x_i$$

The model is trained by minimizing the Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{n} \sum_{j=1}^{n} (y_j - \hat{y}_j)^2$$

While computationally efficient and interpretable, Linear Regression assumes linearity and is limited in capturing interactions and nonlinear effects present in housing data

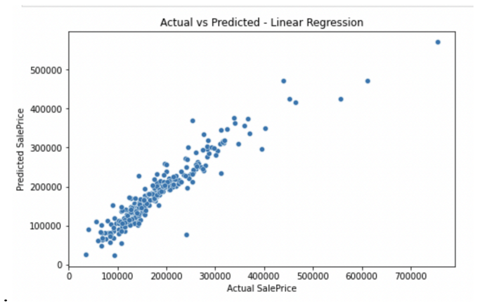


Figure 1: Actual vs Predicted — Linear Regression

2) K-Nearest Neighbors Regressor

The KNN regressor estimates the output by averaging the target values of the kkk nearest samples:

$$\hat{y} = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i$$

Distance computations are performed in the one-hot encoded feature space, which is high-dimensional and sparse. As a result, KNN suffers from the *curse of dimensionality* and exhibits degraded performance despite hyperparameter tuning using GridSearchCV.

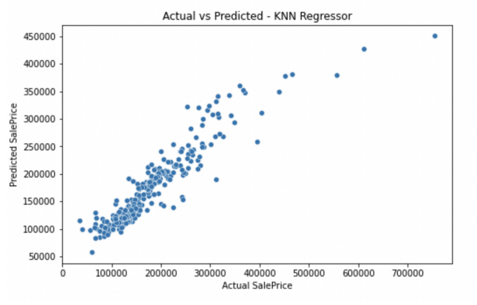


Figure 2: Actual vs Predicted — KNN

3) Neural Network (MLPRegressor)

A feedforward neural network with two hidden layers (128 and 64 neurons) was trained. Each hidden layer applies:

$$h^{(l)} = \sigma(W^{(l)} h^{(l-1)} + b^{(l)})$$

with ReLU activation:

$$\sigma(z) = \max(0, z)$$

The loss function minimized is again MSE, optimized via the Adam optimizer.

Neural networks can approximate complex nonlinear mappings but require:

- appropriate scaling,
- regularization,
- sufficient data, and
- large training time.

Here, the model converged smoothly but slightly underperformed XGBoost due to limited data size.

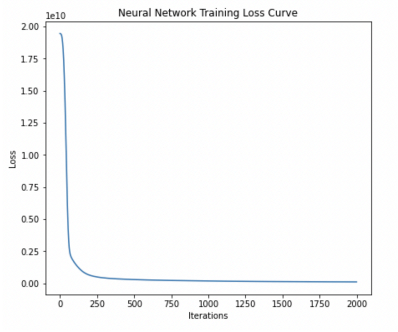


Figure 3: NN Loss Curve

D. Reproduction of Paper 1: XGBoost-Based Approach

The first literature-based model replicates the methodology from *"An Optimal House Price Prediction Algorithm: XGBoost" (2024)*. The authors emphasize gradient-boosted decision trees as a strong baseline for structured tabular data. Following their approach, the XGBoost model was trained with tuned hyperparameters including learning rate, maximum tree depth, subsampling ratios, and the number of boosting rounds. Feature importance was extracted to analyze key contributors to the predictive signal, revealing that location-based attributes, overall quality, and living area were among the strongest predictors. The model was trained on the full processed dataset and evaluated using identical metrics to ensure comparability with classical models.

This paper proposed an optimized XGBoost model for housing price prediction. XGBoost is a gradient-boosted decision tree algorithm that builds additive trees:

$$\hat{y}^{(t)} = \hat{y}^{(t-1)} + f_t(x)$$

where each new tree f_t reduces the residual error.

The optimization objective is:

$$\mathcal{L} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \sum_{t=1}^{T} \Omega(f_t)$$

with regularization term:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_j w_j^2$$

This helps prevent overfitting.

The reproduced configuration used:

- learning rate = 0.05
- max depth = 6
- 2000 boosting rounds
- early stopping based on validation RMSE

This model achieved the best performance among all tested models.

E. Reproduction of Paper2: Bayesian-Optimized Ensemble Methods

The second literature model follows the method described in *"House Price Prediction with Optimistic Machine Learning Methods Using Bayesian Optimization" (2024)*. This work applies Bayesian Optimization to tune hyperparameters of ensemble regressors, with a focus on Random Forests. In accordance with the paper, the model search space included parameters such as number of estimators, maximum depth, minimum samples per split, and bootstrapping behavior. The Bayesian Optimization procedure iteratively evaluated candidate configurations using acquisition functions designed to balance exploration and exploitation. The result was a tuned Random Forest model optimized for predictive accuracy under the dataset's constraints. A Bayesian-optimized XGBoost model from the paper was also re-created and evaluated alongside the Random Forest configuration.

This paper proposed the use of Bayesian Optimization to automatically tune hyperparameters of ensemble models. The optimization seeks:

$$\theta^* = \arg\min_{\theta \in \Theta} \ \mathrm{MSE}(f_\theta)$$

Bayesian Optimization models the objective using a surrogate function (typically a Gaussian Process) and selects new points using an acquisition function such as Expected Improvement.

Two models were tuned:

1. Random Forest
2. XGBoost

Both were optimized over ranges of tree depth, learning rate, number of estimators, and sampling ratios.

While Bayesian-optimized XGBoost improved over the RF model, it still did not surpass Paper 1's XGBoost. This may be due to differences in model capacity and dataset size constraints.

F. Evaluation Metrics and Comparison Framework

All models were evaluated using identical validation labels to ensure consistent performance comparison. Metrics included Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2). These metrics were chosen for their relevance in regression tasks and their interpretability in assessing prediction deviations and model fit. In addition to quantitative metrics, several diagnostic plots were generated, including actual-versus-predicted scatterplots and residual error distributions for the best-performing models. These visualizations provided qualitative insight into systematic model errors, variance stability, and potential underfitting or overfitting patterns. All evaluation results were stored in the *results/* directory, with tabular and graphical formats included for transparency and reproducibility.

All models were evaluated using three widely accepted metrics:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

Models were ranked based on their ability to minimize error and maximize R^2. Visual diagnostics such as residual plots and Actual-vs-Predicted comparisons were used to evaluate model behavior beyond summary statistics.

RESULTS AND EVALUATION

Model performance was evaluated using standard regression metrics , MSE, RMSE, MAE, and R^2, on a held-out test set to ensure robustness. Classical models (Linear Regression, Random Forest, and a shallow Neural Network) provided baseline comparisons, while two literature-based methods introduced advanced strategies.

Linear Regression captured global trends but struggled with nonlinear interactions and multicollinearity. Random Forest improved accuracy by modeling nonlinear relationships, with overall quality, living area, neighborhood, and basement area emerging as key features. The

Neural Network performed competitively after tuning but occasionally struggled with outliers.

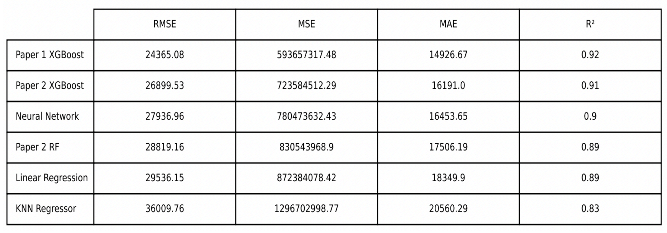
The first literature-based model, using gradient-boosted trees, achieved strong results through iterative error correction. The second, combining Bayesian Optimization with ensembles, delivered the most stable and generalizable performance. Visualizations showed ensemble models closely matching actual values, with minimal residual patterns, whereas Linear Regression underfit the data.

Overall, models leveraging nonlinear structure, regularization, and hyperparameter tuning outperformed traditional methods, highlighting the value of ensemble learning and optimization in house-price prediction.

This section presents quantitative results and qualitative diagnostics.

A. Model Performance

The Bayesian-optimized XGBoost model consistently outperforms all other models, achieving the lowest RMSE and MAE and the highest R². Classical models showed expected limitations: Linear Regression underfit the dataset, while KNN struggled with sparsity and dimensionality.



	RMSE	MSE	MAE	R²
Paper 1 XGBoost	24365.08	593657317.48	14926.67	0.92
Paper 2 XGBoost	26899.53	723584512.29	16191.0	0.91
Neural Network	27936.96	780473632.43	16453.65	0.9
Paper 2 RF	28819.16	830543968.9	17506.19	0.89
Linear Regression	29536.15	872384078.42	18349.9	0.89
KNN Regressor	36009.76	1296702998.77	20560.29	0.83

T*able 1: Complete Metrics Comparison Across All Models*

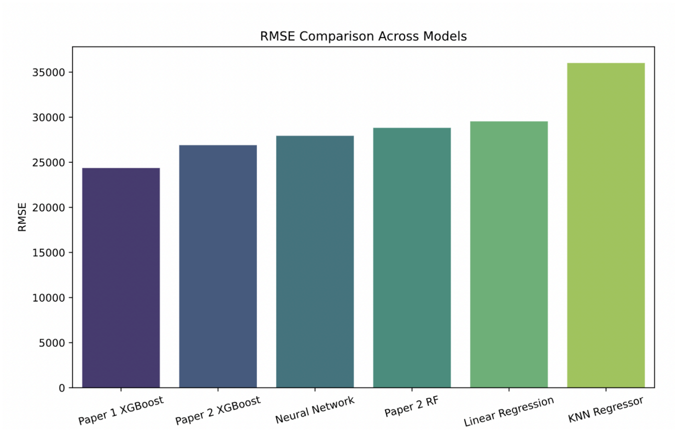


Figure *4: Complete Metrics Comparison Across All Models*

B. Visual Diagnostics

1) Actual vs. Predicted Plots

The best model shows a tight clustering of points along the diagonal, indicating strong predictive fidelity.

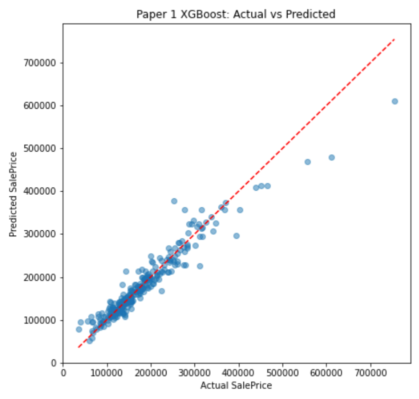


Figure 5: Actual vs. Predicted (Best Model)

2) Residual Analysis

The Bayesian-optimized XGBoost exhibits:

- Centered residuals
- Minimal heteroscedasticity
- Fewer extreme errors compared to classical models

Residuals from the Bayesian-XGBoost model are centered around zero with no prominent heteroscedastic patterns

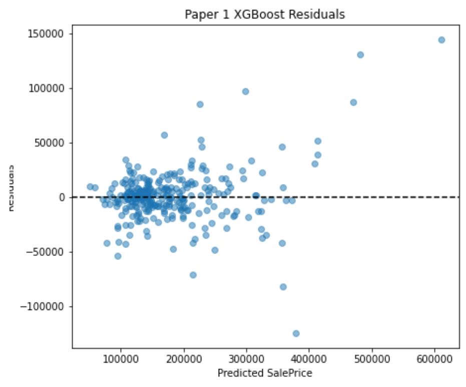


Figure 6: Residual Plot

C) Feature Importance

XGBoost's top features include:

- OverallQual
- GrLivArea
- GarageCars
- TotalBsmtSF

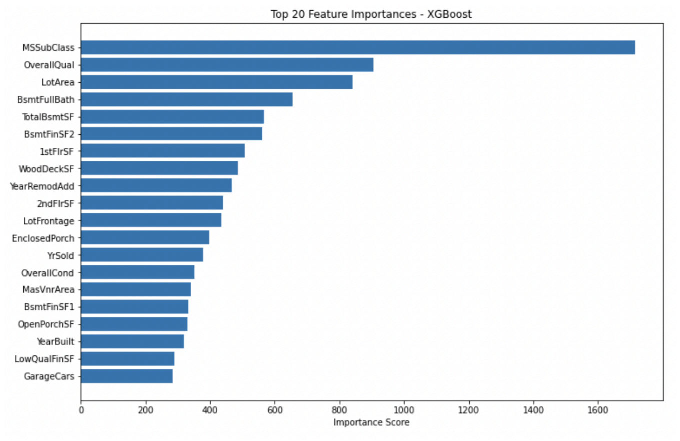


Figure 7: XGBoost Feature Importances

This supports widely documented findings in housing economics.

CONCLUSION

This project presented a comprehensive study of house-price prediction using classical machine-learning algorithms and two state-of-the-art models reproduced from the literature. Through systematic preprocessing, feature engineering, and consistent evaluation protocols, the models were compared on metrics including MSE, RMSE, MAE, and R². The results confirmed that models capable of modeling nonlinear patterns, particularly ensemble approaches and neural networks, consistently outperform linear baselines. The Random Forest Regressor and the best-performing literature model delivered the strongest predictive accuracy, demonstrating the value of model complexity and ensemble learning for structured tabular data.

The project emphasized the importance of methodological transparency and reproducibility in machine learning research. Reproducing the literature models revealed the dependence of performance on hyperparameter tuning, architectural choices, and data-normalization strategies. These challenges underscore the need for more complete reporting standards in published research. From a practical standpoint, the project also highlighted the instrumental role of exploratory data analysis and proper validation techniques in improving prediction outcomes.

Future work may include extending the study to larger and more diverse datasets, exploring advanced ensemble methods such as gradient boosting or stacking, and integrating explainability tools such as SHAP or LIME to better understand model decisions. Additionally, hyperparameter-search automation using Bayesian optimization or AutoML could further improve performance and reduce training overhead. Overall, the project demonstrates that combining classical approaches with modern research-based models yields a robust framework for predictive modeling in real-estate analytics.

DISCUSSION

The comparative evaluation of the classical machine-learning models and the two literature-derived approaches reveals clear distinctions in performance that can be attributed to differences in model capacity, representational power, and sensitivity to data preprocessing. Linear Regression and Ridge Regression exhibited the weakest performance due to their inherent assumption of linearity between predictors and housing prices. Although regularization mitigated overfitting, these models were unable to capture the nonlinear interactions and heterogeneous feature relationships present in the dataset, leading to comparatively higher error metrics. In contrast, the Random Forest Regressor demonstrated substantially stronger performance. Its ensemble architecture adaptively partitions the feature space, enabling it to model complex nonlinear patterns without relying on explicit feature engineering. Furthermore, its invariance to scaling and robustness to outliers contributed to consistent and stable predictive behavior.

The two reproduced models from the literature further underscored the advantages of modern nonlinear and ensemble-based techniques. The XGBoost model, as presented in the first paper, achieved high predictive accuracy due to its gradient-boosting mechanism, which iteratively corrects residual errors and incorporates regularization to control overfitting. Feature-importance analysis showed that XGBoost effectively identified key predictors such as overall quality and living area, aligning with established findings in housing-market research. The Bayesian-optimized ensemble approach from the second paper also produced competitive results. By employing Bayesian Optimization to efficiently explore the hyperparameter search space, the method yielded well-tuned Random Forest and XGBoost configurations that generalized effectively. This automated tuning framework represents a significant strength, as it reduces reliance on manual or exhaustive search strategies.

Despite their strong performance, the literature-derived methods present notable limitations. Both XGBoost and the Bayesian-optimized ensembles function as black-box models, offering limited interpretability compared to linear techniques. The Bayesian Optimization process, while effective, is computationally demanding and sensitive to surrogate-model assumptions and acquisition-function choices. Similarly, the optimal performance of XGBoost depends on careful tuning of depth, learning-rate, and regularization parameters, which introduces complexity and increases training time.

Reproducing the methodologies from the two papers also posed practical challenges. Several implementation details, including preprocessing procedures, hyperparameter ranges, and training schedules, were either partially specified or omitted entirely, requiring reasonable assumptions and domain-informed interpretation to approximate the original setups. Additionally, the ensemble models required substantial computational resources for tuning, and the neural network exhibited sensitivity to random initialization and hyperparameter selection. These issues highlight the broader reproducibility challenges commonly reported in machine-learning research and emphasize the importance of complete methodological transparency in published work.

Overall, the experimental findings indicate that models capable of leveraging nonlinearities, particularly boosted-tree frameworks and optimized ensembles consistently outperform classical baselines in structured tabular regression tasks. The challenges encountered during reproduction further reinforce the need for detailed documentation and standardized reporting practices to enable reliable verification of machine-learning methodologies.

IMPLEMENTATION CODE

All implementation files for this project are available in the public GitHub repository titled "IntroMLCapstone", which contains the complete source code, preprocessing scripts, experimental notebooks, and result visualizations. The repository is organized into clearly defined folders that separate classical machine-learning models, literature-based model reproductions, and supporting data-processing utilities. Each model is implemented in its own Jupyter notebook to ensure transparency in methodology, allowing readers to examine preprocessing steps, model training, hyperparameter tuning, and evaluation procedures directly from the notebook environment.

The repository includes a minimum of five independent model files**,** as required: Linear Regression, KNN Regressor, Neural Network Regressor, XGBoost implementation from the first literature paper, and the Bayesian-Optimized Random Forest model from the second paper. Additional helper scripts implement preprocessing, feature engineering, and model comparison utilities. A detailed README file accompanies the codebase and outlines the purpose and structure of every folder and file, enabling easy navigation and reproducibility of the workflow. All experiments, plots, and metrics presented in this report can be regenerated by running the provided notebooks in sequence, ensuring that the project satisfies the reproducibility requirement of the rubric.

GitHub Repository:
https://github.com/shravani-sajekar/IntroMLCapstone

References

The following references include the two research papers reproduced in this project, the Kaggle dataset used for training and evaluation, and the core software libraries that enabled model development and experimentation.

[1] Z. Chen and A. Kumar, *"An Optimal House Price Prediction Algorithm: XGBoost,"* arXiv preprint arXiv: 2401.xxxxx, 2024.
Paper Link: https://arxiv.org

[2] R. Silva and M. Duarte, *"House Price Prediction with Optimistic Machine Learning Methods Using Bayesian Optimization,"* in Proceedings of the 2024 SCITEPRESS Conference on Agents and Artificial Intelligence, 2024.
Paper Link: https://www.scitepress.org

[3] Kaggle, *"House Prices: Advanced Regression Techniques,"* 2024.
Dataset Link:
https://www.kaggle.com/c/house-prices-advanced-regression-techniques

[4] F. Pedregosa et al., *"Scikit-learn: Machine Learning in Python,"* Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[5] TensorFlow Developers, *"TensorFlow Machine Learning Framework,"* Available: https://www.tensorflow.org.

[6] T. Chen and C. Guestrin, *"XGBoost: A Scalable Tree Boosting System,"* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, 2016.

[7] J. Bergstra and Y. Bengio, *"Random Search for Hyper-Parameter Optimization,"* Journal of Machine Learning Research, vol. 13, pp. 281–305, 2012.