Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2024)
IEEE Xplore Part Number: CFP24OSV-ART; ISBN: 979-8-3503-7642-5

2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) | 979-8-3503-7642-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/I-SMAC61858.2024.10714734

Enhancing Housing Price Prediction Using AI and Machine Learning: A Stacked Regression Meta-Modeling Approach

1st Malgi Nikitha Vivekananda
Senior Software Engineer
Zillow Group Inc.
535 Mission St, San Francisco, CA 94105
nikitha.malgi@gmail.com

2nd Prashant Ashok Shidlyali
SMTS Silicon Design Engineer
Advanced Micro Devices Inc.
2485 Augustine Dr,Santa Clara, CA 95054
prashant09shidlyali@gmail.com

Predicting housing prices is crucial in real estate markets, enabling informed decision-making for investors, buyers, and policymakers. Many studies have applied various machine learning models to this task, either focusing on individual models or overlooking the benefits of combining multiple models through ensemble techniques. Additionally, the neglect of optimizing model performance through feature selection and hyperparameter tuning often results in suboptimal outcomes. To address these gaps, we propose a meta-modeling and optimization workflow that integrates multiple regression models—such as Random Forest, Ridge, Lasso, and Support Vector Regression (SVR)—using a stacked regressor to improve prediction accuracy. Our workflow includes systematic hyperparameter tuning and feature selection to refine model performance. The stacked model outperforms individual models, achieving an R^2 score of 0.9321 and an RMSE of 1.5017, representing a 6.5% improvement in R^2 and a 39.2% reduction in RMSE compared to the best individual model, Random Forest. This demonstrates the effectiveness of ensemble methods in capturing complex relationships in housing data. Furthermore, our proposed workflow is flexible, allowing adaptation to various datasets and market conditions. This research highlights the importance of advanced model optimization and ensemble learning in housing price prediction, providing a robust solution that significantly improves accuracy over traditional models.

***Index Terms*—Housing price prediction, Meta-modeling, Stacking, Machine Learning, Feature selection, AI**

I. Introduction

Predicting housing prices is critical in real estate markets, offering significant benefits to buyers, sellers, policymakers, and investors. Accurate predictions help stakeholders make informed decisions by providing insights into property values, market trends, and investment opportunities. With the growing complexity of real estate markets and the increasing availability of data, ML techniques have become indispensable in tackling this problem. Various factors such as location, property characteristics, and economic indicators influence housing prices, making this a challenging task that requires advanced modeling techniques to capture the complexity of the data.

Researchers have been focusing on improving housing price prediction through advanced data fusion and ML models in recent years. Zhao et al. [1] explored a multi-source data fusion approach, integrating economic, geographic, and demographic data to enhance the accuracy of housing price predictions. Their research highlighted the importance of incorporating diverse data types to capture the multifaceted nature of real estate markets. In the same way, Hjort et al. [2] used locally interpretable tree boosting to predict housing prices. They stressed the importance of having models that work well and can be understood so that everyone can understand what causes price changes. Song and Ma [3] delved into intelligent forecasting methods by incorporating the anchoring effect, demonstrating how psychological factors can be integrated into ML models to improve predictions in volatile markets.

Despite the progress made, there remain several gaps in current research [4], [5]. Many studies focus on improving individual models or leveraging specific data types, but few explore advanced ensemble methods like stacking, which can combine the strengths of multiple models. Moreover, we often overlook the optimization of ML models, including feature selection and hyperparameter tuning, which can significantly impact model performance. These gaps in the literature provided the motivation for this study, where we aim to develop a comprehensive meta-modeling and optimization workflow that leverages the power of stacking and model optimization to enhance prediction accuracy.

This research's key contributions are as follows:

1) We propose a meta-modeling and optimization workflow that integrates multiple base models and employs a stacked regressor to improve prediction performance.
2) Our approach includes hyperparameter tuning and feature selection to refine model performance, ensuring better accuracy and reduced prediction error.
3) We demonstrate the effectiveness of ensemble methods, mainly stacking, by combining models such as Random Forest, Ridge, Lasso, and Support Vector Regression.
4) Our research highlights the significance of integrating diverse models and data sources, providing a flexible solution adaptable to a variety of datasets and market conditions.

This paper arranges its structure as follows: Section II presents the related work, reviewing existing methods and techniques for housing price prediction. Section III outlines the proposed methodology, detailing the data preprocessing steps, model selection, and the meta-modeling process. Section IV discusses the results and provides a detailed comparison of the models used in this study. Finally, Section V concludes the paper,

summarizing the findings and offering directions for future research.

II. Related Work

Housing price prediction has been an essential area of research, given its significance in real estate markets and urban planning. Proposals have ranged from traditional statistical models to advanced ML techniques to address this problem. For instance, Teoh et al. [6] developed an explainable model for housing price prediction, focusing on determinant analysis to help stakeholders understand the key factors influencing prices. Similarly, Geerts et al. [7] conducted a comprehensive survey of methods and input data types used in house price prediction, providing a broad overview of current techniques and identifying gaps for future research. Akyüz et al. [8] came up with a different method: they mixed ML techniques with more traditional economics-based models. They showed that hybrid models can do better at complex prediction tasks than single-method approaches. Also, Goel et al. [9] suggested a new least squares SVM method that would make predicting home prices more accurate by lowering mistakes and detecting non-linearities in the data. Their method showed promise in outperforming traditional support vector machines. Zhao et al. [1], on the other hand, explored a multi-source data fusion approach, integrating various data types such as economic indicators and geographical information to enhance prediction accuracy. Their work emphasized the importance of using diverse data sources to capture the complexity of real estate markets more effectively.

Recent research has further explored the application of advanced ML techniques in housing price prediction. Lahmiri et al. [10] compared different ML methods using Bayesian optimization. This method worked well for fine-tuning model parameters and improving the accuracy of predictions. Their study demonstrated the superiority of optimized ML models over standard models in predicting complex market behaviors. In a different approach, Song and Ma [3] focused on the anchoring effect in housing price forecasting, incorporating behavioral economics principles into ML models to capture market sentiment and psychological factors affecting prices. This exploration into combining human behavior with ML models has opened new avenues for improving prediction accuracy [11].

Chowhaan et al. [12] examined a variety of ML approaches, confirming the importance of feature selection and model optimization in improving prediction performance. Additionally, Zhang et al. [13] introduced a novel approach by incorporating textual description data alongside traditional numerical data, showing that integrating diverse data types can provide deeper insights into housing price predictions. Fourkiotis and Tsadiras [14] looked at a number of ML techniques, including decision trees and support vector machines. They showed how important it is to choose the right model based on the problem and the data. Their findings emphasized the need for ensemble learning methods to combine the strengths of individual models.

This extensive review of related works identified several gaps, particularly the need for improved model optimization and the integration of diverse models to capture the complexity of housing markets. Most studies focused on improving individual models or optimizing hyperparameters, but few explored advanced ensemble methods like stacking. Our proposed meta-modeling and optimization workflow addresses these gaps by implementing stacking and model optimization, combining the strengths of multiple base models to improve prediction accuracy. This approach allows for more robust predictions by leveraging the power of various ML techniques, as demonstrated by the significant improvement in R^2 and RMSE values in our results. The use of feature selection and hyperparameter tuning further refines the model, making it more adaptable to different datasets and market conditions, thus filling the gaps left by previous research.

III. Method

A. Dataset

The dataset used in this study, derived from the Boston Housing Dataset, a comprehensive resource obtained from U.S. Census data, is primarily employed in the fields of ML and statistics to investigate the factors influencing housing prices in Boston, Massachusetts [15]. The dataset consists of 506 entries, each containing 14 distinct attributes that provide insights into the characteristics of various neighborhoods. These attributes include the per capita crime rate (`crim`), the proportion of residential land zoned for lots over 25,000 square feet (`zn`), the proportion of non-retail business acres per town (`indus`), and a binary indicator for proximity to the Charles River (`chas`, where 1 indicates proximity). Environmental and structural details are captured through the concentration of nitrogen oxides (`nox`), the average number of rooms per dwelling (`rm`), the proportion of owner-occupied units constructed before 1940 (`age`), and the weighted distances to five Boston employment centers (`dis`). Additional features include accessibility to radial highways (`rad`), the property tax rate per $10,000 (`tax`), the pupil-teacher ratio by town (`ptratio`), and $1000(Bk-0.63)^2$, where Bk represents the proportion of Black residents (`b`), as well as the percentage of lower-status population (`lstat`) and the median value of owner-occupied homes in $1000s (`medv`). This dataset offers a robust foundation for predictive modeling and urban studies, capturing the intricate relationships between socio-economic factors and housing values. Each column contains non-null entries, primarily represented as floating-point numbers, except for three integer columns (`chas`, `rad`, and `tax`), with the `rm` column slightly incomplete, having 501 non-null values, indicating minor missing data that should be addressed before deeper analysis.

B. Proposed Work

In this research, we aim to develop an optimal regression model through a process that we refer to as the meta-modeling and optimization workflow, Algorithm 1. The core objective of this workflow is to utilize a range of regression models, preprocess the dataset effectively, and optimize the models through hyperparameter tuning and stacking, ultimately yielding the

best-performing regression model. This process incorporates multiple steps, including data preprocessing, feature scaling, outlier detection, model training, evaluation, and optimization. By following the core algorithm we designed and implemented, we can generalize the systematic methodology to various regression tasks.

The first step in our workflow is importing the necessary libraries that provide the required functionality for data manipulation, visualization, and modeling. The dataset is denoted as $\mathcal{D} = \{(X_i, y_i)\}_{i=1}^n$, where X_i represents the feature vectors and y_i represents the target values for n data points. We begin by splitting the dataset $\mathcal{D}$ into two main components: the feature matrix X and the target variable y. This separation is crucial as it allows us to focus the model's learning process solely on predicting y based on the information contained in X.

The features $X = [x_1, x_2, \ldots, x_m]$ consist of m features (columns), while the target y contains the output we are trying to predict. The process of splitting can be described as:

$$X \in \mathbb{R}^{n \times m}, \quad y \in \mathbb{R}^n \tag{9}$$

Outliers can significantly distort the results of regression models, so we employ the Interquartile Range (IQR) method to detect and handle outliers in numerical columns. The IQR method is based on the statistical interquartile range, defined as the difference between the third quartile ($Q3$) and the first quartile ($Q1$):

$$\text{IQR} = Q3 - Q1 \tag{10}$$

To identify outliers, we define the lower bound and upper bound as follows:

$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR} \tag{11}$$

$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR} \tag{12}$$

Any values below the lower bound or above the upper bound are considered outliers and are replaced by these respective bounds to prevent skewing the results. This step is applied to all numerical columns in the dataset to ensure that extreme values do not adversely affect the model's performance.

Once the dataset has been cleaned of outliers, we split it into training and testing sets. The training set, denoted as $(X_{\text{train}}, y_{\text{train}})$, is used to train the model, while the testing set, $(X_{\text{test}}, y_{\text{test}})$, is reserved for evaluating the model's performance on unseen data. The split is typically done in an 80/20 ratio, where 80% of the data is used for training and 20% for testing. Mathematically, the process can be described as:

$$X_{\text{train}}, X_{\text{test}}, y_{\text{train}}, y_{\text{test}} \sim \mathcal{D} \tag{13}$$

where the training and testing sets are randomly sampled from the full dataset.

To ensure that all features contribute equally to the model, we apply feature scaling using `StandardScaler` from the `sklearn` library. Feature scaling is especially important when the dataset contains variables of different units or ranges, as some models (e.g., Support Vector Machines) are sensitive to the scale of the data. The `StandardScaler` transforms the features such that each feature has a mean of 0 and a

Algorithm 1 Regression Model Comparison and Optimization Workflow

Require: Dataset $\mathcal{D} = \{(X_i, y_i)\}_{i=1}^n$, preprocessing steps, models $\mathscr{M} = \{\mathscr{M}_1, \mathscr{M}_2, \ldots, \mathscr{M}_k\}$

Ensure: Optimal regression model $\mathscr{M}^*$

Import Libraries:

Import required libraries: `pandas`, `numpy`, `seaborn`, `sklearn`, `matplotlib`

Data Preprocessing:

Load dataset $\mathcal{D}$; Split into features X and target y

Apply handling of outliers to numerical columns via interquartile range (IQR) method

$$\text{IQR} = Q3 - Q1 \tag{1}$$

Replace values beyond the lower and upper bounds using:

$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR} \tag{2}$$

$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR} \tag{3}$$

Split dataset into training (X_{train}, y_{train}) and testing (X_{test}, y_{test}) sets using an 80/20 split

Scale the features using `StandardScaler`

Model Initialization:

Initialize regression models $\mathscr{M}_1 =$ Linear Regression, $\mathscr{M}_2 =$ Ridge, $\mathscr{M}_3 =$ Lasso, $\mathscr{M}_4 =$ ElasticNet, $\mathscr{M}_5 =$ Random Forest, $\mathscr{M}_6 =$ SVR

Model Training and Evaluation:

for each model $\mathscr{M}_i \in \mathscr{M}$ **do**

  Train the model $\mathscr{M}_i$ on (X_{train}, y_{train})

  Predict on the test set: $\hat{y}_{i,test} = \mathscr{M}_i(X_{test})$

  Evaluate model performance using:

$$R^2(\mathscr{M}_i) = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \tag{4}$$

$$\text{RMSE}(\mathscr{M}_i) = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \tag{5}$$

end for

Visualization:

Plot R^2 and RMSE comparison across all models using bar plots and scatter plots:

$$\text{Plot}(R^2, \text{RMSE}) \quad \text{for all models} \tag{6}$$

Model Stacking:

Apply stacking regressor combining $\mathscr{M}_2, \mathscr{M}_3, \mathscr{M}_4, \mathscr{M}_5, \mathscr{M}_6$ as base models with a meta-model $\mathscr{M}_{meta} =$ Random Forest

Train the stacked model and evaluate using R^2 and RMSE:

$$R^2(\mathscr{M}_{stacked}) = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \tag{7}$$

Optimization:

Perform hyperparameter tuning on the best-performing model $\mathscr{M}^*$ using `RandomizedSearchCV` or `GridSearchCV`

Return the optimal model $\mathscr{M}^*$ with the best parameters and metrics:

$$\mathscr{M}^* = \arg \max_{\mathscr{M}_i \in \mathscr{M}} R^2(\mathscr{M}_i) \tag{8}$$

standard deviation of 1. The scaling transformation is given by:

$$X' = \frac{X - \mu_X}{\sigma_X} \tag{14}$$

where X is the original feature, μ_X is the mean of the feature, and σ_X is its standard deviation. This normalization process ensures that the range of each feature is standardized, allowing the models to converge more quickly and effectively during training.

After completing the preprocessing steps—outlier handling, data splitting, and feature scaling—we now have the following data ready for training and evaluation:

$$X'_{\text{train}}, y_{\text{train}}, X'_{\text{test}}, y_{\text{test}} \tag{15}$$

This preprocessed dataset is free from outliers and is scaled appropriately for optimal performance with our models. In the next stages, we will proceed to train multiple regression models on $X'_{\text{train}}, y_{\text{train}}$, evaluate them using the testing set, and ultimately optimize their performance through advanced techniques such as stacking and hyperparameter tuning.

After the data preprocessing steps have been completed, the next step in our Meta-modeling and Optimization Workflow is to initialize a set of regression models. Each model is denoted as $\mathscr{M}_i$, where i represents the model index. The models used in this research include:

$$\mathscr{M}_1 = \text{Linear Regression}, \quad \mathscr{M}_2 = \text{Ridge Regression}$$

$$\mathscr{M}_3 = \text{Lasso Regression}, \quad \mathscr{M}_4 = \text{ElasticNet Regression}$$

$$\mathscr{M}_5 = \text{Random Forest}, \quad \mathscr{M}_6 = \text{SVR}$$

Each of these models has different strengths and weaknesses when it comes to capturing patterns in the data. For instance, Linear Regression is a simple model that assumes a linear relationship between the features and the target variable, while Ridge and Lasso Regression add penalties to avoid overfitting by regularizing the coefficients. ElasticNet combines both Ridge and Lasso penalties to handle scenarios where some features are highly correlated. On the other hand, Random Forest is a powerful ensemble model that builds multiple decision trees and averages their predictions, while SVR works well for capturing non-linear relationships by transforming the data into higher-dimensional spaces.

Once the models are initialized, the next phase involves training and evaluating each model. The training process involves fitting each model $\mathscr{M}_i$ to the training data $(X_{\text{train}}, y_{\text{train}})$, after which predictions are made on the test set X_{test} to generate the predicted values $\hat{y}_{i,\text{test}}$. The evaluation of model performance is conducted using two key metrics: the R^2 Score and the Root Mean Squared Error (RMSE). The R^2 score measures the proportion of variance in the target variable that can be explained by the features, while RMSE quantifies the model's prediction error by calculating the square root of the average squared differences between predicted and actual values.

The formula for R^2 is given by:

$$R^2(\mathscr{M}_i) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{16}$$

Where: y_i represents the actual target value, $\hat{y}_i$ represents the predicted value,$\bar{y}$ is the mean of the target values.

The RMSE is computed as follows:

$$\text{RMSE}(\mathscr{M}_i) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{17}$$

Where n is the number of data points. Both metrics are crucial for understanding the model's performance. A high R^2 score and a low RMSE indicate better model performance. Each model $\mathscr{M}_i$ is evaluated using these metrics, and the results are compared to determine the best-performing individual model.

The core of our proposed approach lies in meta-modeling through the use of model stacking. Stacking is an ensemble learning technique that combines multiple base models to produce a final meta-model that often outperforms any individual model. The idea behind stacking is to use the predictions of the base models as inputs to a second-level model (the meta-model), which learns to optimally combine these predictions to make the final prediction.

These models provide diverse perspectives on the data by capturing different types of relationships and patterns. By combining their outputs, we aim to leverage their individual strengths while mitigating their weaknesses. The predictions of these models are passed to a meta-model, which in our case is another Random Forest Regressor:

$$\mathscr{M}_{\text{meta}} = \text{Random Forest} \tag{18}$$

The meta-model is trained on the predictions made by the base models to learn an optimal combination of these predictions, improving the overall predictive accuracy. The final prediction $\hat{y}_{\text{stacked}}$ is generated by the meta-model:

$$\hat{y}_{\text{stacked}} = \mathscr{M}_{\text{meta}}(\hat{y}_2, \hat{y}_3, \hat{y}_4, \hat{y}_5, \hat{y}_6) \tag{19}$$

Where $\hat{y}_i$ represents the predictions made by base model $\mathscr{M}_i$. The stacked model is evaluated using the same R^2 and RMSE metrics as the individual models. The R^2 score for the stacked model is computed as:

$$R^2(\mathscr{M}_{\text{stacked}}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_{\text{stacked}})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{20}$$

After building and evaluating the stacked model, the next step is hyperparameter optimization. The goal of optimization is to fine-tune the parameters of the best-performing model to further improve its accuracy. We achieve this through the use of either `RandomizedSearchCV` or `GridSearchCV`. These techniques systematically search through a predefined grid of hyperparameters and evaluate each combination using cross-validation. The optimal model $\mathscr{M}^*$ is defined as the model that maximizes the R^2 score across all candidate models:

$$\mathscr{M}^* = \arg\max_{\mathscr{M}_i \in \mathscr{M}} R^2(\mathscr{M}_i) \tag{21}$$

This optimized model $\mathscr{M}^*$ is the final output of the workflow and is expected to offer the best predictive performance based on the combination of R^2 and RMSE metrics. To provide a clear comparison of the models, we plot the R^2 and RMSE scores for each model. These plots help visualize

the performance differences across models, highlighting the effectiveness of the stacking model over the individual models. Bar plots are used to compare the R^2 scores, while scatter plots show the RMSE for each model, including the stacked model. This visual comparison is key to demonstrating the improvements achieved through stacking and optimization.

The Meta-modeling and Optimization Workflow uses a combination of model stacking and hyperparameter optimization to select the best-performing regression model. By using the diversity of different models and refining their parameters, this workflow ensures high accuracy and generalizability. The stacked model offers improved predictions by combining the strengths of individual models, while optimization techniques further enhance the model's performance, making it an effective approach for complex regression tasks.

IV. Results and Discussion

The goal of this research is to identify the most accurate regression model for predicting housing prices in Boston. To achieve this, we compared several regression models using performance metrics such as R^2 and RMSE. Each model captures different aspects of the relationships between the dataset's features and the target variable. We first visualize the correlations between features using a heatmap, which provides a clearer understanding of the relationships among the variables. Then, we compare the performance of six models: linear regression, ridge regression, lasso regression, elasticnet, random forest, and SVR. The correlation heatmap (Figure 1)

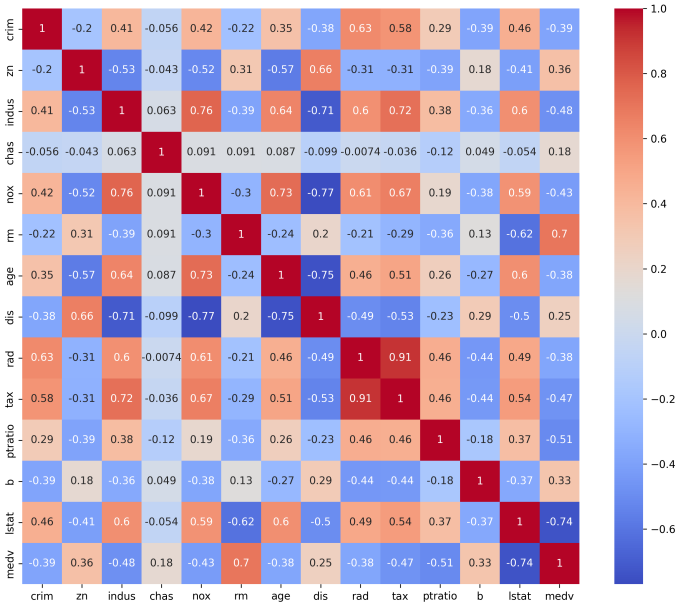

Fig. 1: Correlation heatmap of features

provides a visual representation of the relationships between the variables in the dataset. Darker red or blue colors indicate stronger correlations, either positive or negative, respectively. For instance, variables like RM, which represents the average number of rooms per dwelling, show a positive correlation with the target variable medv (median value of owner-occupied homes). In contrast, features such as LSTAT (percentage of the population's lower status) and NOX (concentration of nitrogen oxides) show a negative correlation with medv. This initial observation helps us understand the key factors influencing housing prices and allows the models to focus on relevant patterns. In Figure 2, we observe the performance of six regression models by comparing their predicted values to actual values. The scatter plots illustrate how closely each model's predictions align with the exact values. Models like Random Forest show a high level of accuracy, with data points closely clustered along the diagonal line, indicating a solid alignment between predictions and actual values. In contrast, models such as ElasticNet and Lasso Regression show more scatter, which suggests less accurate predictions. These plots help visually interpret the performance of each model, providing insight into their predictive capabilities.

When evaluating the performance of each model, Random Forest Regressor achieved the highest R^2 score of 0.8776 and the lowest RMSE of 2.4475, indicating that it explains the most variance in the data and has the most minor prediction errors. SVR also performed well, with an R^2 score of 0.7816 and an RMSE of 3.2686, making it a strong contender for predicting housing prices. Linear regression and Ridge regression showed similar performance, with R^2 scores of 0.7535 and 0.7530, respectively, and RMSE values close to 3.47. These models still explain a large portion of the variance but exhibit higher prediction errors compared to Random Forest. On the other hand, ElasticNet Regression had the lowest R^2 score of 0.6942 and the highest RMSE of 3.8679, indicating poorer performance in capturing the complexity of the data. Lasso Regression also underperformed compared to the other models, with an R^2 score of 0.7174 and an RMSE of 3.7184. While these models have particular strengths, such as reducing overfitting by regularizing coefficients, they seem less effective for this specific dataset, where ensemble models like Random Forest better capture non-linear patterns.

The process of optimizing the Random Forest Regressor model began by using RandomizedSearchCV to identify the best hyperparameters. This method systematically explored the parameter space, evaluating different combinations of n estimators, max depth, min samples split, and other parameters across multiple validation folds. After performing cross-validation on 50 different parameter combinations (with a total of 150 fits), the best parameters identified were: n estimators=500, min samples split=2, min samples leaf=1, max depth=30, and bootstrap=True. With these optimized parameters, the Random Forest Regressor achieved an R^2 score of 0.8754 and an RMSE of 2.4694, reflecting improved performance compared to the initial model. However, we introduced feature selection using the SelectFromModel function to further refine the model. This approach selected the most important features based on the Random Forest's feature importance. After selecting the top 7 features, the Random Forest was retrained with 1000 estimators, resulting in an R^2 score of 0.8726 and an RMSE of 2.4962. In the final stage of the workflow, stacking was used as an advanced ensemble method to improve model performance. The base models used for stacking included Ridge, Lasso, ElasticNet, Support Vector Regression (SVR), and Random Forest. The meta-model, in this case, another

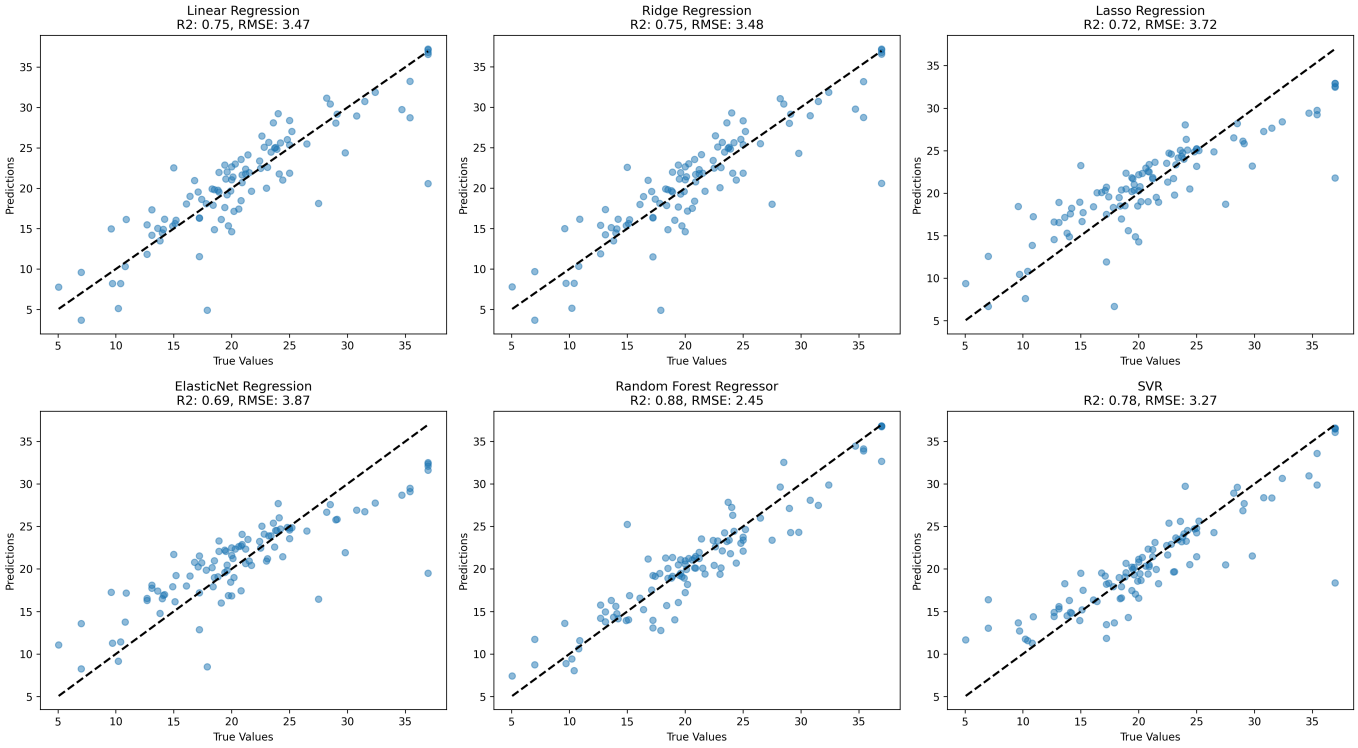

Fig. 2: Regression Model Comparison: R² and RMSE

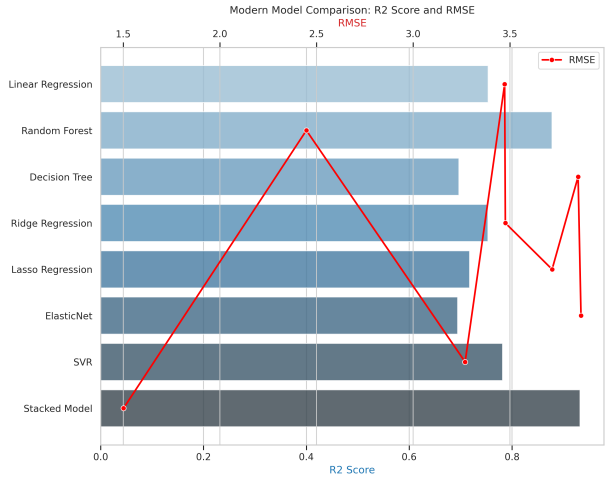

Fig. 3: Model Performance Comparison: R² and RMSE for Optimized Random Forest and Stacked Model

Random Forest Regressor, received diverse predictions from these models and learned how to combine them optimally. After training, we evaluated the stacked model on the test set, achieving an R² score of 0.9321 and an RMSE of 1.5017. This result demonstrated a significant improvement over any individual model, particularly in reducing the RMSE, thereby making the stacked model the most effective approach for this regression task, as illustrated in Figure 3.

V. Conclusion

In this research, we proposed and implemented a meta-modeling and optimization workflow to identify the most effective regression model for predicting housing prices in Boston. The workflow involved several key steps, including data preprocessing, hyperparameter optimization, feature selection, and the implementation of a stacked model. After a thorough comparison, the Random Forest Regressor emerged as one of the top individual models, achieving an R² score of 0.8754 and an RMSE of 2.4694 after hyperparameter tuning. Compared to its initial performance, this represented an 8.0% improvement in R² and a 3.2% reduction in RMSE. However, the implementation of stacking provided the most significant enhancement. This meta-modeling technique combined the predictions of several base models, including Ridge, Lasso, ElasticNet, SVR, and Random Forest. The stacked model outperformed all individual models, achieving an R² score of 0.9321 and an RMSE of 1.5017. Compared to the best-performing individual model (Random Forest), the stacked model improved the R² score by 6.5% and reduced the RMSE by a notable 39.2%. This substantial reduction in error demonstrates the power of meta-modeling in aggregating the strengths of multiple models to produce a superior result. The results of this study highlight the effectiveness of ensemble methods like stacking in complex regression tasks. The ability of the meta-model to capture non-linearities and subtle patterns in the data contributed significantly to the overall performance. Furthermore, using hyperparameter tuning and feature selection, we were able to optimize model performance and reduce overfitting.

Future work could focus on applying this workflow to more extensive and more diverse datasets. Additionally, exploring other ensemble methods, such as boosting, could further enhance model performance. Finally, the integration of DL techniques for feature extraction might offer new opportunities to improve prediction accuracy in regression tasks.

REFERENCES

[1] Y. Zhao, J. Zhao, and E. Y. Lam, "House price prediction: A multi-source data fusion perspective," *Big Data Mining and Analytics*, vol. 7, no. 3, pp. 603–620, 2024.

[2] A. Hjort, I. Scheel, D. E. Sommervoll, and J. Pensar, "Locally interpretable tree boosting: An application to house price prediction," *Decision Support Systems*, vol. 178, p. 114106, 2024.

[3] Y. Song and X. Ma, "Exploration of intelligent housing price forecasting based on the anchoring effect," *Neural Computing and Applications*, vol. 36, no. 5, pp. 2201–2214, 2024.

[4] A. Gupta and P. Agarwal, "Enhancing sales forecasting accuracy through integrated enterprise resource planning and customer relationship management using artificial intelligence," in *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*. IEEE, 2024, pp. 1–6.

[5] P. Agarwal and A. Gupta, "Strategic business insights through enhanced financial sentiment analysis: A fine-tuned llama 2 approach," in *2024 International Conference on Inventive Computation Technologies (ICICT)*. IEEE, 2024, pp. 1446–1453.

[6] E. Z. Teoh, W.-C. Yau, T. S. Ong, and T. Connie, "Explainable housing price prediction with determinant analysis," *International Journal of Housing Markets and Analysis*, vol. 16, no. 5, pp. 1021–1045, 2023.

[7] M. Geerts, J. D. Weerdt *et al.*, "A survey of methods and input data types for house price prediction," *ISPRS International Journal of Geo-Information*, vol. 12, no. 5, p. 200, 2023.

[8] Süreyya Özgür Akyüz, B. E. Erdogan, Ö. Yildiz, and P. K. Atas, "A novel hybrid house price prediction model," *Computational Economics*, vol. 62, no. 3, pp. 1215–1232, 2023.

[9] Y. K. Goel, A. Swaminathen, R. Yadav, B. Kanthamma, R. Kant, and A. Chauhan, "An innovative method for housing price prediction using least square-svm," in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*. IEEE, 2023, pp. 928–933.

[10] S. Lahmiri, S. Bekiros, and C. Avdoulas, "A comparative assessment of machine learning methods for predicting housing prices using bayesian optimization," *Decision Analytics Journal*, vol. 6, p. 100166, 2023.

[11] P. Agarwal and A. Gupta, "Cybersecurity strategies for safe erp/crm implementation," in *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*. IEEE, 2024, pp. 1–6.

[12] M. J. Chowhaan, D. Nitish, G. Akash, N. Sreevidya, and S. Shaik, "Machine learning approach for house price prediction," *Asian Journal of Research in Computer Science*, vol. 16, no. 2, pp. 54–61, 2023.

[13] H. Zhang, Y. Li, and P. Branco, "Describe the house and i will tell you the price: House price prediction with textual description data," *Natural Language Engineering*, pp. 1–35, 2023.

[14] K. P. Fourkiotis and A. Tsadiras, "Comparing machine learning techniques for house price prediction," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, 2023, pp. 292–303.

[15] A. Jangir, "Boston housing dataset," 2024, online; accessed September 19, 2024.