Project 4: House Price Regression

Mark Herrmann *Team Leader, Algorithm Design*
Joseph Hooser *Feature and Label Analysis, Preprocessing*
Jason Powell *Implementation* Johntae Leary *Documentation*

May 21, 2025

Abstract

This report summarizes a project focused on predicting housing prices using different regression models. This project aimed to develop an accurate predictive model through Kaggle's "House Prices: Advanced Regression Techniques" dataset, to assist stakeholders in the housing market. The methodology involved data preprocessing, including handling missing values, One-Hot Encoding, and Standard Scaling, and creating new features. Multiple regression models, including Linear Regression, Ridge, Lasso, Support Vector Regression (SVR), Random Forest, XGBoost, and a Neural Network, were trained and their hyperparameters tuned using GridSearchCV. An ensemble model averaging the predictions of the top performers was also evaluated. Model performance was assessed using Root Mean Squared Error (RMSE), R-squared (R^2), and Mean Absolute Error (MAE). Cross-validation was used for evaluating the top-performing model(SVR). The results indicate that SVR achieved the lowest RMSE and highest R^2 on the validation set, outperforming all other models, with a low training time. This project successfully delivered a predictive model capable of estimating house prices with an average error (RMSE) around $20,688 over a 200 k-fold cross validation.

1 Introduction

The housing market is a complex and volatile environment where prices are influenced by a multitude of factors. Accurate prediction of housing prices is invaluable for buyers, sellers, realtors, and investors, allowing them to make informed, data-driven decisions. This project addresses the challenge of predicting housing prices by leveraging machine learning regression algorithms.

Our primary objective was to predict housing prices based on property features using a housing dataset on Kaggle. An accurate predictive model that can help stakeholders navigate the market more effectively. Our approach involved training and evaluating multiple regression models on a preprocessed dataset, tuning their hyperparameters to optimize performance, identifying the best-performing model, and analyzing the results.

2 Literature Review

Our work is informed by existing research in regression models used for housing prediction. A notable paper with a similar goal is "House price prediction based on different models of machine learning" by Ni Chuhan (March 2024). This study compared the performance of common regression algorithms on a house price prediction task using a Kaggle dataset, similar to the one we utilized.

The significance of Ni Chuhan's study lies in its comparison of algorithms, allowing us to anticipate which regressors might perform well on housing datasets (e.g., XGBoost, Random Forest) and providing benchmark error metrics to aim for. Their study provided a valuable baseline and insight into the applicability of various techniques to this problem domain.

3 Dataset Overview

The dataset used for this project is sourced from the Kaggle competition "House Prices: Advanced Regression Techniques" (`https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data`).

- **Source:** Kaggle
- **Training Set:** 1,460 Rows, 81 Features
- **Validation Set:** 20% of the original training data
- **Target Variable:** SalePrice (Numerical, continuous)

The dataset includes a mix of numerical and categorical features describing various aspects of residential homes in Ames, Iowa.

4 Methodology

Our methodology followed a standard machine learning pipeline, focusing on data preparation, model training and tuning, model selection, evaluation, and cross-validation of the top-performing model.

4.1 Data Preprocessing

Data quality is paramount for building accurate models. Our preprocessing steps addressed missing values and prepared features for modeling.

4.1.1 Handling Missing Values

Missing values were imputed based on the nature of the feature:

- **Numerical Fixes:**
  - *LotFrontage*: Filled using the median value of the respective neighborhood group, assuming proximity implies similar lot frontage characteristics.
  - *GarageYrBlt*: Missing values were assumed to be the same year the house was built ('YearBuilt').
  - *MasVnrArea*: Missing values were filled with 0, assuming no masonry veneer if the area was not recorded.
- **Categorical Fixes:**
  - *Electrical*: The single missing value was filled with the most common value (mode).
  - Other features with 'NA' indicating the absence of a feature (e.g., PoolQC, MiscFeature, Alley, Fence, FireplaceQu, GarageQual, etc.) were filled with the string "None".

4.1.2 Feature Encoding and Scaling

- **Encoding:** Categorical features were converted into a numerical format suitable for machine learning algorithms using One-Hot Encoding. This process expanded the feature space from 81 to over 150 features.
- **Scaling:** Numerical features were scaled using 'StandardScaler' to ensure that features with different ranges do not disproportionately influence models sensitive to scale.

4.2 Feature Engineering

To improve model performance and capture more meaningful relationships, we engineered three new features:

- **Total Square Feet ('TotalSF'):** Sum of above-ground living area and total basement area: 'GrLivArea + TotalBsmtSF'.
- **Age of House ('HouseAge'):** The difference between the year sold and the year built: 'YrSold - YearBuilt'.
- **Age of a Remodel ('RemodelAge'):** The difference between the year sold and the year of the last remodel: 'YearRemodAdd' - 'YrSold'.

4.3 Data Split

The preprocessed and engineered training dataset was split into training and validation sets, with 80% used for training the models and 20% reserved for evaluating their performance.

4.4 Model Selection and Training

We trained a suite of regression models, including:

- Linear Regression
- Ridge Regression
- Lasso Regression
- Support Vector Regression (SVR)
- Random Forest Regressor
- XGBoost Regressor
- Neural Network

4.4.1 Hyperparameter Tuning

Hyperparameters for each model were tuned using `GridSearchCV` to find the optimal configuration that minimized prediction error on the validation set. The final chosen parameters based on tuning were:

- **SVR:** `kernel='poly'`, `C=1000`, `epsilon=1000`, `gamma='scale'`, `degree=3`, `coef0=3`
- **Ridge:** `alpha=10`
- **Lasso:** `alpha=100`, `max_iter=5000`
- **Random Forest:** `n_estimators=150`, `max_depth=30`, `min_samples_split=5`, `min_samples_leaf=1`
- **XGBoost:** `n_estimators=1000`, `learning_rate=0.1`, `max_depth=3`
- **Neural Network:** Architecture of two hidden layers (64 and 32 units) with ReLU activation, trained with Adam optimizer and early stopping based on validation loss.

4.4.2 Ensemble Method

An ensemble model was created by averaging the predictions of the top 6 performing individual regressors (excluding Linear Regression).

4.5 Evaluation Metrics

Model performance was evaluated using standard regression metrics:

- **Root Mean Squared Error (RMSE):** $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$. Measures the average magnitude of the errors. Sensitive to large errors.
- **R-squared (R^2):** $1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$. Represents the proportion of the variance in the dependent variable that is predictable from the independent variables. A value closer to 1 indicates a better fit.
- **Mean Absolute Error (MAE):** $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$. Measures the average magnitude of the errors without considering their direction. Less sensitive to outliers than RMSE.

5 Experimental Results

We evaluated each trained model on the validation set using the specified metrics. The training time for each model was also recorded.

5.1 Model Performance Comparison

The performance metrics for each model on the validation set are summarized in the table below:

Table 1: Model Performance Comparison on Validation Set

Model	RMSE	R-Squared	MAE	Time (seconds)
SVR	25644	0.914	14830	1.0
XGBoost	26753	0.907	16587	15.06
Ensemble Regressor	27376	0.902	15844	178.57
Random Forest	30089	0.882	17928	9.09
Ridge Regression	30604	0.878	18957	0.39
Lasso Regression	31256	0.873	18770	0.41
Neural Network	32851	0.859	20205	152.63
Linear	83215	0.097	23965	1.2

Visual comparisons using bar charts were generated to illustrate the relative performance across RMSE, R-squared, MAE, and Training Time.

5.2 Model Performance Comparison

The performance metrics for each model on the validation set are summarized in the table below:

Table 2: Model Performance Comparison on Validation Set

Model	RMSE	R-Squared	MAE	Time (seconds)
SVR	25644	0.914	14830	1.0
XGBoost	26753	0.907	16587	15.06
Ensemble Regressor	27376	0.902	15844	178.57
Random Forest	30089	0.882	17928	9.09
Ridge Regression	30604	0.878	18957	0.39
Lasso Regression	31256	0.873	18770	0.41
Neural Network	32851	0.859	20205	152.63
Linear	83215	0.097	23965	1.2

5.3 Comparison Plots

To visually compare the performance across different models, bar charts were generated for RMSE, R-Squared, Mean Absolute Error, and Training Time. These visualizations highlight the relative strengths and weaknesses of each model.

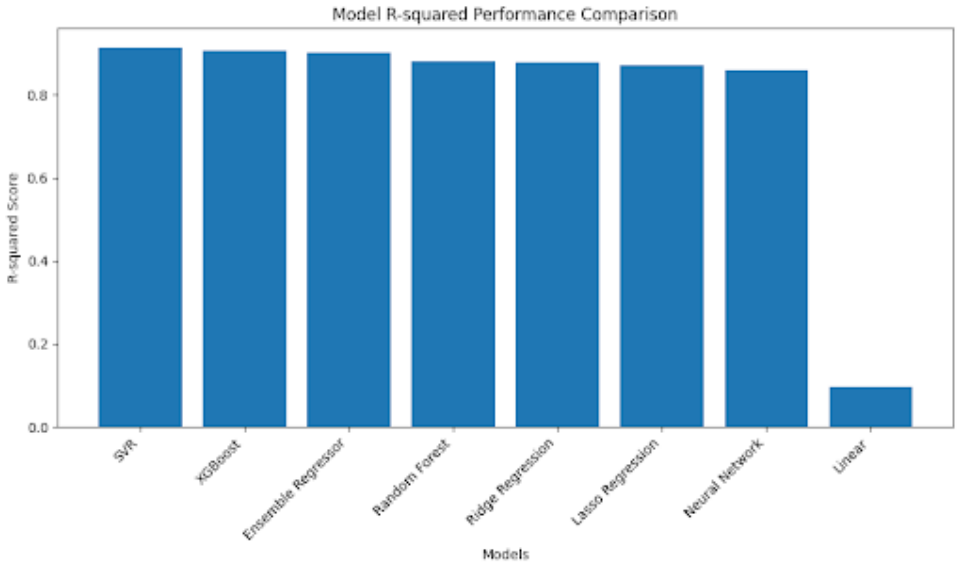

Figure 1: Comparison of Model RMSE Scores

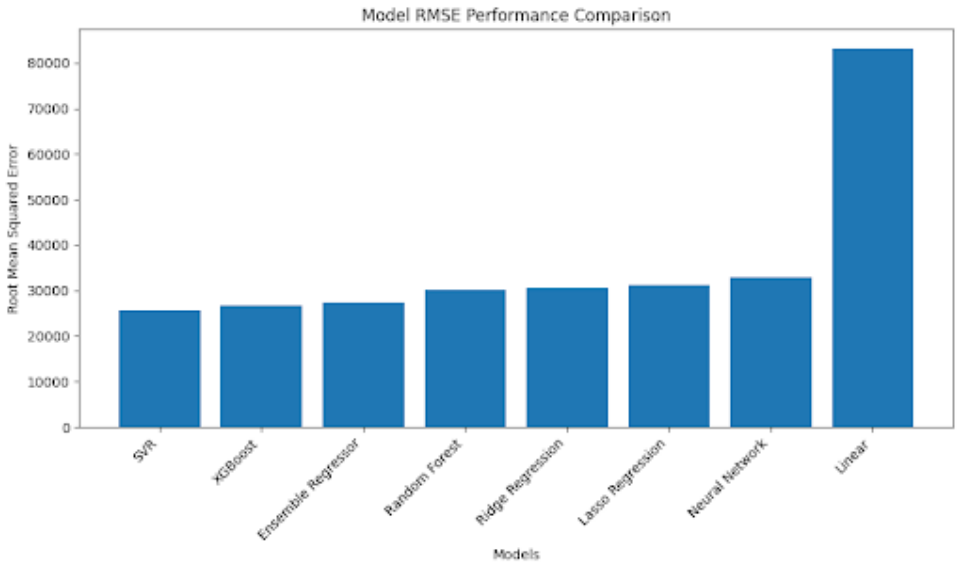

Figure 2: Comparison of Model R-Squared Scores

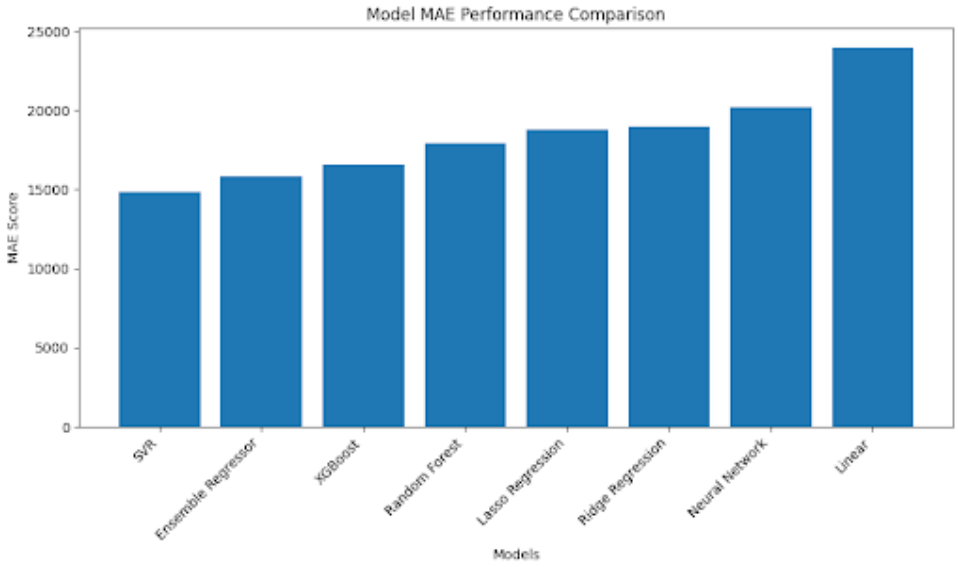

Figure 3: Comparison of Model Mean Absolute Error Scores

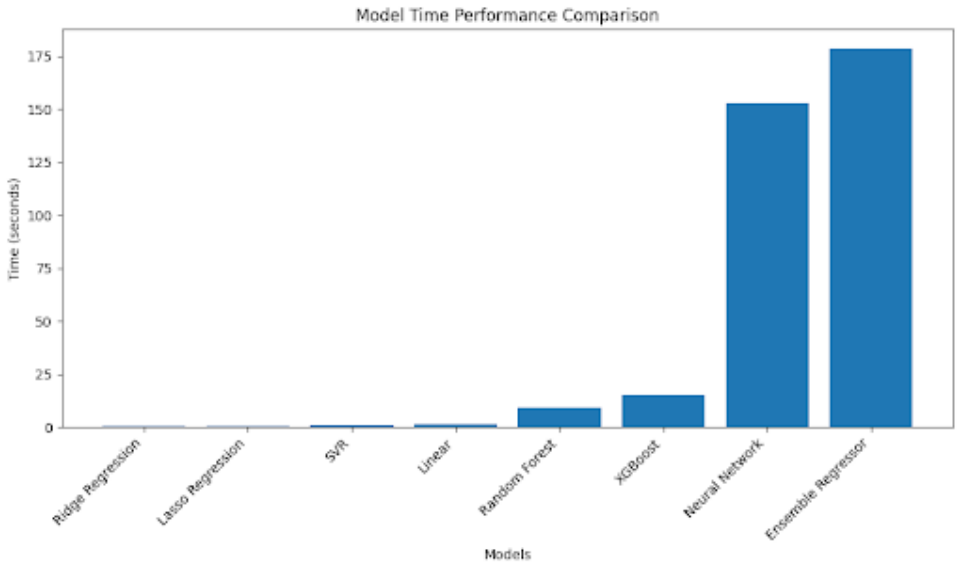

Figure 4: Comparison of Model Training Times

5.4 Cross-Validation Results for SVR

To obtain a more reliable estimate of the SVR model's performance and assess its variability, we performed cross-validation on the training data with varying numbers of folds.

Table 3: SVR Cross-Validation Performance

Folds	Mean RMSE	Mean R-Squared	Mean Absolute Error	Runtime (seconds)
20	26,749	0.823	15,688	13
50	24,362	0.774	15,510	34
200	20,788	0.665	15,167	157

The cross-validation results show that as the number of folds increases, the Mean RMSE and MAE generally improve, suggesting a more robust performance estimate. However, the Mean R-Squared value decreases. This observation could imply that models with higher bias might perform better when evaluated across more subsets of the data, reducing variance within the data subsets. The runtime increases significantly with the number of folds, as expected due to the model being trained many times.

5.5 Discussion of Results

Based on the validation set, SVR performed the best across RMSE, R-squared, and MAE metrics. It also exhibited a relatively fast training time compared to the ensemble methods like XGBoost and Random Forest, which also performed well but were computationally more intensive. The Ensemble Regressor showed competitive performance, demonstrating the potential benefit of combining multiple models, but high training time makes it less practical for rapid iteration and deployment without significant computational power. Ridge and Lasso performed reasonably well after tuning, with Lasso and Ridge showing significant improvement over simple Linear Regression. The Neural Network's performance was slightly worse than the top models but it still performed well, however, its training time was too long.

The mean house price in the training set was approximately $180,921.20 with a standard deviation of $79,442.50. An RMSE of around $24,612 on the validation set with 1105 support vectors or $20,788 (SVR 200-fold CV mean) represents a significant reduction in error compared to simply predicting the mean price, indicating the models capture substantial variance in the data.

6 Discussion of Technical Aspects

6.1 Existing Breakthroughs and Challenges

House price prediction is a widely studied problem in machine learning. Existing breakthroughs often involve applying advanced tree-based ensemble methods (like Gradient Boosting Machines, Random Forests, XGBoost, LightGBM), deep learning techniques (Neural Networks), and feature engineering tailored to real estate data. Regularization techniques (Lasso, Ridge) are also standard for handling high-dimensional data.

Challenges in existing studies and this domain include:

- **Data Quality:** Missing values, outliers, and inconsistencies are common.
- **Feature Engineering:** Identifying and creating features that properly capture market dynamics and property value is crucial but challenging.
- **Market Volatility:** Housing prices are influenced by macroeconomic factors, interest rates, and local developments not captured in static datasets.
- **Geographic Specificity:** Models trained on data from one area may not generalize well to others.
- **Interpretability:** Complex models like ensembles or neural networks can be difficult to interpret, making it hard to understand *why* a prediction was made.

6.2 Overall Model Flowchart

The overall process flow for our project can be described sequentially:

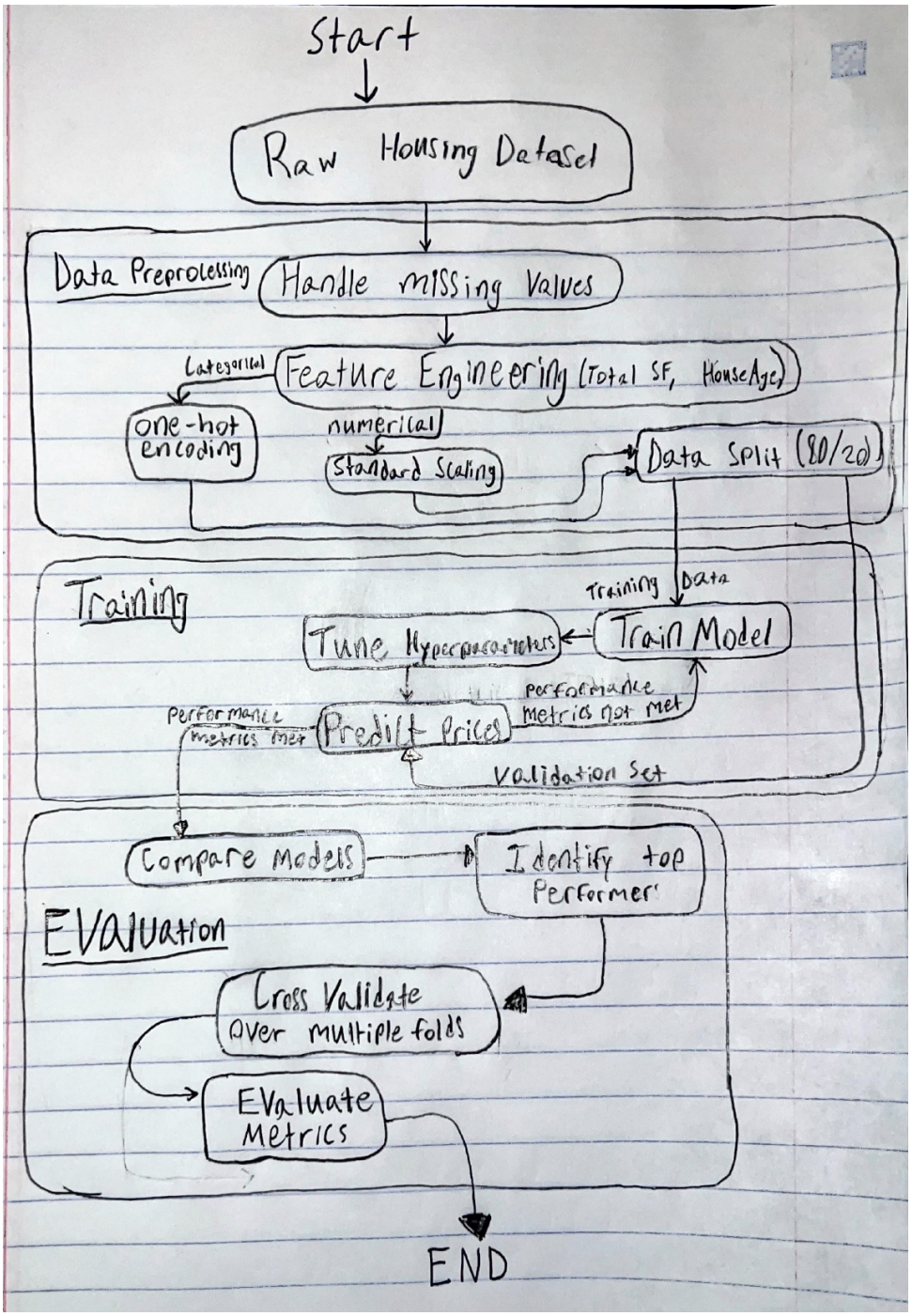

Figure 5: Project Flow

6.3 Cross-Validation Performance

As reported in Table 3, the SVR model's performance was evaluated using k-fold cross-validation on the training data with k=20, 50, and 200 folds. The results showed that increasing the number of folds generally led to lower mean RMSE and MAE, suggesting that the model generalizes slightly better when evaluated across a larger number of diverse training/validation splits. However, the decrease in Mean R-Squared with more folds is counterintuitive for a model aiming to capture variance; this might indicate that with very fine-grained splits, the model potentially becomes slightly more biased or that the R-squared metric is more sensitive to variance differences in smaller folds compared to error magnitudes (RMSE/MAE). It could also suggest that the specific hyperparameters tuned on the single 80/20 split are not perfectly optimal across all possible cross-validation folds, highlighting the trade-off in tuning on a single split versus cross-validation. Cross-validation provides a more reliable estimate of model performance than a single train-validation split by reducing the dependence on a particular random division of the data.

6.4 Confusion Matrix, AUC, Precision, Recall, F1-score

As stated in the evaluation metrics section, Confusion Matrix, AUC (Area Under the Receiver Operating Characteristic Curve), Precision, Recall, and F1-score are metrics specifically designed for *classification* problems, where the goal is to predict a discrete class label.

Our project is a *regression* problem, where the goal is to predict a continuous numerical value (SalePrice). Therefore, these classification metrics are not applicable or meaningful in this context. Our evaluation focused entirely on the appropriate regression metrics: RMSE, R-squared, and MAE.

6.5 Training, Validation, and Test Curves and Addressing Overfitting/Underfitting

Observing learning curves (plots of performance metrics like loss or RMSE over training iterations/epochs for both the training and validation sets) is a standard practice to diagnose overfitting and underfitting.

- **Underfitting:** Occurs when the model is too simple to capture the underlying patterns in the data.
- **Overfitting:** Occurs when the model learns the training data too well, including noise, and performs poorly on unseen data.

For the Neural Network, we used early stopping based on validation loss to combat overfitting by ending training early when validation performance starts to degrade. Hyperparameter tuning via GridSearchCV also implicitly addresses these issues by selecting parameters that perform best on validation data, thus balancing bias and variance.

6.6 Running Time

The training times for each model on the validation split are reported in Table 2 and visually compared in the training time bar chart. The times range from very fast (Linear, Ridge, Lasso, SVR) to moderate (Random Forest, XGBoost) to significantly slower (Neural Network), with the Ensemble having the highest cumulative time. These times reflect the computational complexity of the algorithms. SVR offered a training speed that was among the top models of 1 secong.

6.7 Team Contributions

This project was completed through the collaborative efforts of the team members, with specific roles assigned to manage different aspects of the project:

- **Mark Herrmann (Team Leader, Algorithm Design):** Led the overall project direction, coordinated team activities, facilitated discussions on strategy and model selection, and played a key role in designing the approach to model training and evaluation.
- **Joseph Hooser (Feature and Label Analysis, Preprocessing):** Responsible for understanding the dataset features, identifying the target variable, performing the crucial data cleaning steps (handling missing values), and preparing the data for modeling through encoding and scaling.

- **Jason Powell (Implementation):** Focused on translating the methodology into code. This included implementing the preprocessing steps, feature engineering, setting up and running the various regression models, performing hyperparameter tuning using GridSearchCV, and executing the evaluation scripts to generate performance metrics.
- **Johntae Leary (Documentation):** Tasked with compiling and structuring the project documentation. This involved synthesizing the work of the team into a coherent report format, describing the methodology, results, and conclusions, and ensuring all project requirements were addressed in the documentation.

Each member contributed to discussions, problem-solving, and ensuring the project progressed effectively towards its objectives.

6.8 Contributions to Existing Studies

Our project builds upon existing studies in house price prediction, including insights from Ni Chuhan's paper. Our key contributions include:

- **Identification of SVR as a Top Performer:** We found that Support Vector Regression, with appropriate hyperparameter tuning, achieved the best performance (lowest RMSE, highest R2) on this specific housing dataset among the individual models tested, potentially offering an alternative top-tier model compared to the often-cited tree-based ensembles like XGBoost and Random Forest in similar studies.
- **Improved Performance Benchmark:** We were able to achieve a lower RMSE (around $20,788 in 200-fold CV mean, compared to the literature's lowest reported RMSE of 30,013) through our comprehensive preprocessing, feature engineering, and rigorous hyperparameter tuning process, demonstrating the impact of careful data preparation and model optimization.
- **Detailed SVR Hyperparameter Tuning:** We specifically highlighted and tuned SVR's parameters, demonstrating that this model can achieve state-of-the-art results in this domain when properly configured.

7 Model Limitations and Considerations

It is important to acknowledge the limitations and considerations of our developed models:

- **Assumption of Pattern Consistency:** The regressors assume that future housing data will follow similar patterns and relationships observed in the training data.
- **Market Volatility:** The housing market is inherently volatile and subject to rapid changes influenced by factors not present in the dataset (e.g., economic recessions, sudden shifts in local demand or supply).
- **Macroeconomic Factors:** Our models do not account for broader macroeconomic shifts, changes in interest rates, or government policies that significantly impact housing prices.
- **Geographic Specificity:** The dataset is from Ames, Iowa. The trained models are likely only reliable for predicting prices within this specific geographic area and may not generalize well to other markets with different characteristics.
- **Outliers and Rare Types:** The models may underperform when predicting prices for rare or unusual house types or properties that are significant outliers in terms of features or price.
- **Overfitting Risk with Unbalanced Categories:** Although One-Hot Encoding was used, highly unbalanced categorical features could potentially lead to overfitting, especially with models sensitive to dimensionality. Further analysis on feature importance and regularization might be needed.

8 Future Development

Based on our findings and limitations, several areas for future development exist:

- **Further Hyperparameter Tuning:** More extensive and systematic tuning of the best models (SVR, XGBoost, Random Forest) with a broader range of hyperparameter values or more advanced tuning techniques could yield further performance improvements.
- **Advanced Feature Engineering:** Exploring interaction terms between more features, incorporating polynomial features, or exploring external data sources (e.g., local economic indicators, school district performance, planned infrastructure projects) could enhance predictive power.
- **Outlier Analysis and Handling:** A more thorough analysis of potential outliers in features and the target variable and applying specific outlier handling techniques (e.g., Winsorizing, robust scaling, using models less sensitive to outliers) could improve robustness.
- **Deployment:** Developing a system or API to deploy the trained model would allow for practical use in making predictions on new, unseen property data.
- **Exploring Other Models:** Investigating other advanced regression techniques or ensemble strategies (e.g., stacking, boosting variants like LightGBM) could potentially lead to better results.

9 Conclusion

In this project, we successfully implemented, tuned, and evaluated multiple regression models for predicting house prices using the Kaggle Ames, Iowa dataset.

We now understand the importance of data preprocessing and feature engineering in preparing the dataset for modeling. While ensemble methods like XGBoost and Random Forest are known for their strong performance, our experiments showed that tuned Support Vector Regression achieved the best performance metrics (lowest RMSE and highest R2) on our validation set while having a favorable training time. The ensemble method also yielded competitive results but at a much higher computational cost.

Based on cross-validation results, our SVR model demonstrated a mean RMSE of approximately \$20,788 (with 200 folds), indicating the model can estimate house prices with an average error within this range on unseen data from the same distribution. This level of accuracy represents a successful outcome for the project, providing a valuable tool for house price estimation within the defined scope and limitations.