A Comparative Study of Selected Machine Learning Methods for Predicting Housing Prices

Kwanele Natha Mnisi

18164049

STK795 Research Report

Submitted in partial fulfillment of the degree

BCom(Hons) Statistics and Data Science

Supervisor(s):Dr P Nagar, Co-supervisor(s):Prof A Bekker

Department of Statistics, University of Pretoria

20 October 2023

Abstract

Comparing machine learning techniques for house price prediction is of paramount importance for selecting accurate and efficient models, thereby enhancing decision-making in real estate transactions and advancing the field of predictive analytics. This study aims to address this critical need by conducting a comprehensive comparison of three prominent machine learning techniques: Random Forest, Support Vector Regression, and XGBoost, with a specific focus on their predictive performance for house price estimation. The primary objective involves a twofold comparison strategy. Firstly, an internal comparison is executed within each technique, where the predictive capabilities of full models and reduced models (featuring feature selection) are contrasted. Secondly, an external comparison is performed across the three techniques to determine the optimal approach for achieving accurate predictions. Results stemming from this investigation unveil a remarkable predictive prowess across all three models. Among the methodologies assessed, XGBoost emerges as the top performer, exhibiting the highest predictive accuracy. Further analysis within the XGBoost framework highlights the efficacy of the reduced model with feature selection in yielding the most favorable outcomes. Interpretation of the findings reveals a consistent trend across all three techniques: the reduced model consistently outperforms its full counterpart. This pattern underscores the pivotal role of feature selection in elevating model performance. Importantly, the results strongly advocate for the incorporation of feature selection processes, as they positively contribute to the predictive precision of the models. This study not only contributes to the selection of suitable machine learning techniques for house price prediction but also underscores the instrumental role of feature selection in enhancing predictive accuracy. The insights presented herein serve as a valuable guide for researchers and professionals seeking reliable predictive models in the dynamic realm of real estate.

Keywords: housing prices, machine learning, prediction, random forest, support vector regression, XGBoost

Declaration

I, *Kwanele Natha Mnisi,* declare that this essay, submitted in partial fulfillment of the degree *BCom(Hons) Statistics and Data Sciences,* at the University of Pretoria, is my own work and has not been previously submitted at this or any other tertiary institution.

_____

Kwanele Natha Mnisi

_____

Priyanka Nagar (Supervisor), Andriette Bekker (Co-Supervisor)

_____

Date

Acknowledgements

I would like to express my gratitude to Tomorrow Trust for their generous financial support. Additionally, I extend my heartfelt appreciation to my supervisor, Dr. Nagar, and my co-supervisor, Prof. Bekker, for guiding and directing me throughout my research journey. Futhermore, the use of the publicly available dataset for this study was conducted under ethics approval number NAS116/2019. I am grateful for the opportunity to access and analyze this data, which has contributed significantly to the quality of this study.

Contents

List of Figures

List of Tables

1 Introduction

Buying or selling a house is one of the most significant financial transactions that most people undertake in their lifetime. For this reason, it is crucial to have an accurate understanding of the value of a property, whether you are a buyer looking to make a wise investment, or a seller seeking to maximize your profit. House pricing is the process of determining the worth of residential properties by taking into consideration several factors including location, amenities, condition and the size of the house(27). External factors such as government policies, supply and demand, economic conditions and interest rates also have an impact on determining the price of a house(38).

House pricing affects a wide range of stakeholders, including estate agents, homeowners, mortgage lenders, renters, investors, local governments and the economy as a whole. For example, changes in house prices can have various impacts such as affecting the availability and cost of mortgage credit, the wealth and mobility of homeowners, the affordability and stability of housing for renters, and the profitability of real estate investments. The fluctuations in the prices can also have spillover effects on consumer spending, construction activity and stability. Potential buyers, sellers and investors frequently asks themselves a multitude of questions when it comes to buying, selling or investing in a house. Questions such as whether they are paying or charging a fair price, or if the timing is optimal for their purchase, sale or investment in a property. Therefore, it is crucial to make accurate predictions about house prices as it can provide insights into when to make a transaction, how much to list or buy a property and how to negotiate a fair deal. Furthermore, house price prediction is crucial for monitoring the housing market, conducting research on real estate markets and it can also help policymakers and regulators to track the housing market and address problems such as affordability, stability and accessibility(29).

In recent years, the use of machine learning algorithms for house price prediction has gained popularity, as it offers an accurate, data-driven approach to estimating a property's value. Machine learning (ML) is a sub-field of artificial intelligence (AI) that involves the scientific study of algorithms and statistical models enabling computer systems to learn from data, recognize patterns, and make decisions without explicit programming(26). There are three main types of machine learning namely, supervised learning, which involves training a model on a labelled data set , unsupervised learning, involving training a model on an unlabelled data set, and reinforced learning, involving an agent learning to make decisions through trial and error. Examples of how the three types of machine learning can be used in the context of house price prediction: Supervised learning can be used to predict the sale price of new, unseen properties based on their features; unsupervised learning can be used for clustering similar properties together based on their features, which can help with market segmentation and identifying trends in the housing market; and, reinforcement learning can be used to optimize the pricing strategy of a real estate agent or broker.

There are various sophisticated machine learning algorithms such as linear regression, decision trees, random forest, support vector regression, XGBoost and neural network that can provide accurate prediction of the value of a property. Winky et al. (14) found that, in terms of predictive powers (using mean absolute error (MAE) and the root mean squared error (RMSE) as performance measures), random forest and gradient boosting machine achieved better performance when compared to support vector machine. Victor et al. (11) built price prediction models using neural networks and decision trees. Between the two algorithms, they found that neural networks algorithm performed better by producing lower mean errors than decision trees. Uzut and Buyrukoglu (37) compared Random forest, gradient boosting and linear regression models. Those models were built by taking different sizes of test sets. The best result was obtained by gradient boosting regression when the size of the test set is 20% with a mean absolute error of was 3.93 for this approach. However, this study will delve into the fascinating world of house price prediction by considering the prediction of sales price using supervised learning techniques, specifically, random forest, support vector regression and XGBoost. The Ames Housing dataset will be used to implement the above mentioned techniques. In addition, this study will also explore the principles behind these advanced techniques and how they can be used to make better decision in the real estate market.

2 Literature Review

In the forthcoming section, an elaborate discussion will be presented for each proposed technique, comprising a comprehensive analysis of their respective advantages, disadvantages, strategies for enhancing their performance, and the corresponding R packages employed for their implementation.

2.1 Random Forest

Random forest is a versatile and powerful machine learning algorithm that utilizes an ensemble learning method to build multiple decision trees and combine their predictions for more accurate results (7). This approach offers several advantages for data analysis and prediction tasks across different domains. Firstly, it can handle high-dimensional data effectively (7), while also being less prone to overfitting than other methods (4). Additionally, random forest provides measures of variable importance, which can be used for feature selection (25). It can handle missing values and maintain accuracy even when a large proportion of the data is missing (36). Moreover, random forest can be used for both classification and regression problems (7).

However, even though random forest is a widely used machine learning algorithm, it does have several limitations that should be taken into account. Firstly, random forest can be computationally expensive and may require a large amount of memory, particularly when working with larger datasets (7). Additionally, it may be more difficult to interpret than other models and produce biased variable importance estimates in some situations (36). For certain types of data, such as sparse data, other algorithms may perform better than random forest (7). Despite these limitations, random forest remains a valuable tool for many applications, and its benefits should be weighed against its drawbacks when choosing an appropriate method for a given task.

The performance of random forest can be improved through various measures and techniques. One effective approach for improving the performance of random forest models is to optimize the hyperparameters of the algorithm, such as the number of trees, the number of features considered at each split, and the minimum size of terminal nodes. Probst et al. (31) suggested that careful hyperparameter tuning is an important step in optimizing the performance of random forest models. The authors found that even small changes to hyperparameter settings could lead to noticeable improvements in model performance. They also noted that different tuning strategies can lead to different optimal hyperparameter settings, indicating that the choice of tuning strategy is an important consideration when optimizing model performance. However, there may be cases where hyperparameter tuning has a relatively small impact on model performance, particularly in cases where the default hyperparameters perform well.

Another way to improve the performance of random forest is to use feature selection techniques to identify and select the most relevant features for the model. Kursa and Rudnicki(22) have demonstrated that feature selection techniques, specifically the Boruta package (23), can improve the performance of random forest by identifying and selecting the most relevant features for the model. This approach has been particularly effective in dealing with high-dimensional data, where selecting the most relevant features can significantly reduce the computational burden and prevent overfitting.

Finally, using a larger and more diverse training dataset can be beneficial, especially when working with a large dataset. Genuer et al. (12) suggest that a diverse training dataset can help to reduce overfitting and improve the LightGBM ability to generalize. Additionally, the authors recommend using parallel computing techniques to speed up the training process, handling missing data appropriately to prevent bias in the model, and employing cross-validation to optimize the parameters of the random forest algorithm and prevent overfitting.

There are several R packages that can be used to implement the Random Forest algorithm in R, including:

Table 1: Summary of popular random forest packages in R

Package	Description
randomForest	This is one of the most widely used packages for random forests in R. It provides a flexible and easy-to-use interface for building and tuning random forest models(33).
ranger	This package is a fast implementation of random forests that can handle large datasets. It uses parallel computing to speed up the model building process(40).
party	This package provides an implementation of random forests and other tree-based methods for regression and classification. It also provides methods for model selection and visualization(15).
RandomForestSRC	This package is specifically designed for survival analysis and can handle high-dimensional data. It also includes methods for variable selection and feature importance(16).
boruta	This package implements a feature selection algorithm for random forests that can identify important variables even in the presence of noise and correlation(22).
caret	This is a comprehensive package for machine learning that includes an implementation of random forests. It also provides tools for preprocessing data, tuning models, and evaluating performance(21).

2.2 Support Vector Regression

According to Drucker et al (10), support vector regression (SVR) is a method that uses the support vector machine (SVM) algorithm to perform regression analysis. While SVM is an algorithm mainly used for classification problems, it can also be extended to regression problems by adjusting the loss function. The primary objective of SVR is to identify the function that best approximates the relationship between the input and output variables. This is done by transforming the original data into a higher-dimensional space using a kernel function, which allows the algorithm to detect non-linear relationships between the input and output variables (10). The algorithm then proceeds to identify a hyperplane that maximizes the margin between the predicted values and the actual values. The margin is defined as the distance between the hyperplane and the closest data points, also known as support vectors. SVR aims to minimize the loss function while ensuring that the predicted values fall within the margin and satisfy a user-specified tolerance value (10).

SVR is a powerful machine learning technique that has gained significant attention in recent years due to its ability to handle complex data and produce accurate predictions (41). It has several advantages over other regression techniques. One of the primary advantages of SVR is its ability to handle non-linear data. According to Zhang and O'Donnel(41), SVR uses kernel functions to transform the original feature space into a higher-dimensional space where the data can be more easily separated, allowing for the handling of non-linear data. Additionally, SVR is robust to noisy data and can handle outliers better than other regression methods, as it focuses on the support vectors that lie closest to the decision boundary.

Moreover, SVR offers a range of kernel functions to choose from, which can be tailored to the specific problem, providing greater flexibility in modeling. Also, it is less prone to overfitting than other regression methods, such as linear regression, because it uses a regularization parameter to control the complexity of the model (34). Drucker et al. (10) further explain that the optimization problem that SVR solves has a unique solution, which means that the resulting model is stable and not dependent on initial conditions. In summary, SVR is a valuable tool for regression problems, especially when dealing with complex, non-linear data with noisy features.

Smola and Schölkopf (34) highlight several disadvantages of support vector regression, such as its computational complexity, sensitivity to kernel function choice, difficulty in interpretation with non-linear kernel functions, and the potential for overfitting with complex models or small training datasets. However, these issues can be addressed using techniques such as kernel selection and regularization to prevent overfitting. Therefore, it is important to use support vector regression appropriately while understanding its limitations.

There are several techniques that can be used to improve the performance of the model. One way to improve performance is through using multiple kernel. Najafzadeh et al. (30) proposed a new algorithm for water quality parameter estimation using multiple-kernel support vector regression (MK-SVR). The study found that the proposed algorithm, which uses a combination of different kernels, outperforms traditional SVR and other regression models in terms of accuracy and generalization performance.

Feature selection is another technique that can be used to improve SVR model performance. The study conducted by Karasu et al. (18) highlights the importance of feature selection in improving the performance of SVR models. They proposed a new forecasting model that employs a wrapper-based feature selection approach using multi-objective optimization techniques for predicting crude oil prices. The wrapper-based approach selects the most relevant features from the input dataset and uses them to train the SVR model, resulting in improved prediction accuracy. This approach is particularly useful in dealing with high-dimensional data, where selecting

the most relevant features can help reduce model complexity and prevent overfitting. The study demonstrates that incorporating feature selection techniques into SVR models can lead to more accurate and robust predictions

Ensemble methods have been found to be effective in enhancing the accuracy and robustness of SVR models. These methods are known for their ability to reduce variance and improve the generalization performance of models, making them suitable for addressing the problem of overfitting commonly observed in SVR models. In a study by Ahmad et al.(1), the authors proposed using ensemble methods based on decision trees, such as random forest and gradient boosting, to improve the accuracy of photovoltaic (PV) power generation prediction models. The experiment results showed that the proposed ensemble methods outperformed traditional SVR models in terms of accuracy and robustness, demonstrating the effectiveness of ensemble methods in improving the performance of SVR models.

There are several packages in R that implement the SVR algorithm, including:

Table 2: Summary of popular SVR packages in R

Package	**Description**
e1071	This package is a popular choice for implementing Support Vector Machines (SVMs) and Support Vector Regression (SVR). It provides a range of functions for building and tuning SVM orSVR models (28).
kernlab	This package provides a wide range of kernel functions for SVMs and SVR. It also includes methods for model selection and cross-validation (19).
caret	This is a comprehensive package for machine learning that includes an implementation of SVR. It provides tools for data preprocessing, model tuning, and performance evaluation(21).
liquidSVM	This is a high-performance SVM/SVR implementation that uses a combination of cutting-edge optimization algorithms and parallel processing to build models quickly and accurately(35).
mgcv	This package implements Generalized Additive Models (GAMs) which can be used for regression tasks. It also includes methods for regularization and smoothing that can be used to build more flexible models than standard SVR(39).

2.3 XGBOOST

XGBoost, also known as eXtreme Gradient Boosting, is a popular open-source software library for gradient boosting. This framework builds an ensemble of weak models to create a more robust and accurate final model. Its design aims to improve the accuracy and speed of machine learning algorithms, particularly in the areas of supervised learning, regression, and classification. As a result, XGBoost has become a go-to algorithm for many data scientists due to its ability to handle large datasets and its high predictive power.

Chen and Guestrin (5) describe XGBoost as an optimized distributed gradient boosting library that is highly efficient, flexible, and portable. It can handle both categorical and continuous data and has a built-in feature selection mechanism. One of the main advantages of XGBoost is its speed and scalability. It uses parallel processing and can take advantage of distributed computing to improve its speed and scalability. This has made it a popular choice for large-scale machine learning applications, such as intrusion detection systems (2).

Another advantage of XGBoost is its high predictive power. It has been shown to have high accuracy in various machine learning tasks, including classification, regression, and ranking. In a study by Ke et al. (20), XGBoost was found to perform better in terms of accuracy and speed than other gradient boosting algorithms. This has made XGBoost a go-to algorithm for many data scientists and machine learning practitioners.

XGBoost also has a built-in feature selection mechanism that can automatically identify the most important features in a dataset. This is particularly useful in applications where the number of features is large and identifying the relevant ones is challenging. Additionally, XGBoost uses regularization techniques to prevent overfitting, making it less prone to errors and more robust.

However, there are also some limitations of XGBoost. One potential issue is overfitting, which can occur if the hyperparameters are not optimized correctly. This can lead to poor performance in practical applications (2). Additionally, the XGBoost algorithm is relatively complex and may require expertise to implement and optimize for specific applications.

Like other machine learning algorithms, XGBoost's performance can be improved by taking certain measures. Optimizing the hyperparameters is a key step in improving the performance of XGBoost. Chen and Guestrin (5) suggest using a combination of grid search and random search to tune the hyperparameters of XGBoost. In particular, they recommend focusing on the learning rate, maximum depth, and gamma values (the minimum loss

reduction required to make a split in the tree) to improve the accuracy of the model and prevent overfitting. By fine-tuning these parameters, it is possible to achieve better results and ensure that the model does not overfit.

In addition to optimizing hyperparameters, feature engineering is another key step in improving the performance of XGBoost. Ke et al. (20) suggest using a combination of feature engineering and XGBoost to achieve better results. Techniques such as one-hot encoding, feature scaling, and binning can be used to create more informative features that can help the algorithm better capture patterns in the data. By preprocessing the data in this way, it is possible to extract more meaningful information from the input variables, and improve the accuracy of the model.

Ensemble methods can also be employed to improve the performance of XGBoost. Bhati et al. (2) suggest that combining XGBoost with other machine learning algorithms through ensemble methods can enhance its predictive power. Specifically, they propose using an ensemble of decision trees along with XGBoost to improve accuracy and reduce variance. By combining the strengths of both XGBoost and decision trees, it is possible to build a more robust and accurate model. This approach is particularly effective when dealing with complex datasets that may require a more diverse range of models to capture all the underlying patterns.

Moreover, distributed computing can significantly enhance the performance of XGBoost, as recommended by Chen and Guestrin (5). The authors suggest employing a cluster of machines to run XGBoost in parallel, thereby reducing the time required for training and prediction. By using distributed computing, the computational load can be shared across multiple machines, enabling the algorithm to process vast amounts of data more efficiently. This approach can lead to significant improvements in training and prediction times, making XGBoost a more powerful tool for large-scale data analysis.

Listed below are several R packages that can be used to implement the XGBoost algorithm:

Table 3: Summary of popular XGBoost packages in R

Package	**Description**
xgboost	The official R package for XGBoost. Provides an efficient and flexible interface for building and tuning XGBoost models. Includes functions for feature selection, cross-validation, and parallel processing (6).
caret	A comprehensive package for machine learning that includes an implementation of XGBoost. Provides tools for data preprocessing, model tuning, and performance evaluation. The caret package also includes several other popular machine learning algorithms (21).
mlr	A machine learning framework that includes an interface for XGBoost. Provides a range of functions for preprocessing data, building models, and evaluating performance. The mlr package also includes other popular machine learning algorithms and methods for ensemble modeling (3).
h2o	Provides an interface for the H2O machine learning platform, which includes an implementation of XGBoost. The H2O platform has a built-in XGBoost function that can be used for various machine learning tasks. It provides a range of features such as automatic feature engineering and model interpretation (24).

2.4 Performance Evaluation

Performance evaluation is a fundamental aspect of machine learning, as it allows for the comparison of various models. When evaluating the performance of a machine learning model, there are several commonly used measures that can be employed. For regression tasks, the mean squared error (MSE) is a commonly used metric that measures the average of the squared differences between the predicted and actual values of the target variable(5). The formula to calculate MSE is as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where n is the number of data points in the dataset, y_i is the observed value of the i^{th} data point and $\hat{y}_i$ is the predicted value of the i^{th} data point.

Another useful performance measure is the root mean squared error (RMSE), which provides an estimate of the standard deviation of the errors by taking the square root of the MSE. The formula to calculate the RMSE is:

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

Additionally, other commonly used performance measures include the mean absolute error (MAE), which measures the average absolute difference between the predicted and actual values of the target variable. The formula to calculate the MAE is as follows:

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

and the R-squared (R^2), which measures the proportion of variance in the dependent variable explained by the independent variables (5) is calculated by the following formula:

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

where $\bar{y}$ is the mean of the observed values.

Random forest, SVR, and XGBoost are popular machine learning techniques that use these metrics to evaluate their performance. Therefore, in this study, these performance measures (MSE, RMSE, MAE and R^2) will be used to compare and analyze the performance of the three proposed techniques.

3 The Ames Housing Dataset

Ames is a city located in Iowa, USA, and the Ames housing dataset examines details of homes sold in Ames between 2006 and 2010, encompassing a range of residential and non-residential structures such as stand-alone garages, condos, and storage areas. This dataset was prepared by Ames Housing Authority, a public organization that caters to the people of Ames, Iowa.

The initial dataset was made up of 113 variables describing 3970 property sales while accounting for nominal, ordinal, continuous, and discrete variables used to calculate assessed values(8). It presented physical measurements of properties alongside computation variables used by the city's assessment process. To simplify the dataset for use by a broad range of users, Dean De Cock (8) parsed out any variables relying on specialized knowledge or previous calculations, leaving only 80 variables featuring details of physical attributes of residential properties. These variables chiefly provide information a typical home buyer would seek regarding a potential property.

The explanatory variables consist of 23 nominal variables, they identify characteristics of the property and dwelling type or structure, 23 Ordinal variables, they are related to the rankings of the quality or condition of rooms and lot characteristics, 20 continuous variables, they are related to the measurements of the area dimensions for each observation, and 14 discrete variables, they are related to the number of bedrooms, bathrooms, kitchen etc and some are related to profiling properties and neighbourhood (8). However, multiple observations such as non-residential and older sales observations were removed, resulting to the final data set containing only 2930 observations.

3.1 Data Cleaning and Exploration

In this sub section, the Ames Housing dataset undergoes preparation and analysis. In section 3.1.1 the focus centers on the treatment of missing values through removal, replacement, and imputation strategies to enhance dataset reliability. Following this, in section 3.1.2 an in-depth analysis uncovers underlying trends and correlations within the Ames housing market dataset. Full variable descriptions are provided in Appendix A.

3.1.1 Data Cleaning

The Ames Housing dataset presents a considerable number of missing values, with 27 variables containing such observations. Nevertheless, various approaches exist to address these instances of missing data within the dataset. These approaches encompass:

1. Removal of missing values: One can opt to eliminate rows or columns with missing values from the dataset by utilizing functions like na.omit() or complete.cases().

2. Replacement of missing values: Another viable option involves substituting missing values with appropriate replacements based on the nature of the data. For instance, numerical values can be replaced with the mean or median of the respective column, while categorical values can be substituted with the mode.

3. Utilization of imputation techniques: Alternatively, more advanced imputation techniques can be employed to impute missing values. The R package mice offers dedicated functions for performing imputation.

The dataset exhibits missing values in various variables (*see Appendix A for variable discription*) , with the top five variables being *Alley*(99%), *Fireplace Qu* (96%), *Pool QC* (93%), *Fence* (80%), *Misc Feature* (48%), and *Lot Frontage* (16%). The absence of certain features can be effectively represented using descriptive categories, such as "no alley access" (NAA), "no fireplace" (NFP), "no pool" (NP), and "no miscellaneous features" (NMF) for the first four variables.

However, replacing missing values in the Lot Frontage variable with descriptive categories is impractical due to its numerical nature. Additionally, the meaning behind the missing values remains unknown, posing a challenge for appropriate value assignment. To address this, the utilization of imputation techniques becomes essential for estimating the missing values.

The *Lot Frontage* variable is influenced by various other variables associated with the physical characteristics and layout of the properties, including *lot area*, *lot shape*, *lot config*, *street*, *neighborhood*, and *MS zoning*. Considering these interdependencies, the mice package (42) in R, which implements multiple imputation methods, was employed to estimate the missing values in the *Lot Frontage* variable.

The mice package facilitates the imputation process by implementing multiple imputation methods, allowing for the estimation of missing data based on observed relationships within the available data (42). By considering the aforementioned variables and their interdependencies, the mice package enables the estimation of missing values in the lot frontage variable. This approach ensures a comprehensive and reliable estimation of the lot frontage, compensating for the lack of initially available information.

Regarding the *garage yr built* variable, the presence of null values signifies that the properties do not have garages. Since substituting these values with meaningful information is not feasible, a dummy variable named *Has Garage* is created. This variable takes the value 0 to represent "no garage" and 1 to indicate "garage present."

When the *Total Bsmt SF* variable has a value of zero, it indicates the absence of a basement in the property. To address the corresponding null values in related basement variables such as *bsmt qual*, *bsmt cond*, *bsmt exposure*, *bsmtfin type 1*, and *bsmtfin type 2*, these null values are substituted with the code "ND," which represents "No Basement."

Null values in other variables are simply omitted from the analysis. Specifically, 32 observations, accounting for approximately 1% of the dataset, have been excluded due to missing values. After effectively handling missing values, the cleaned dataset comprises 82 variables and 2898 observations. This data cleansing process involved utilizing imputation techniques and addressing null values to ensure a comprehensive and reliable dataset for further analysis.

3.1.2 Data Exploration

This section presents a concise yet insightful analysis of the dataset, unveiling valuable patterns and insights that serve as a robust foundation for further investigation and decision-making.

The mean sale price of houses in a real estate market provides essential insights into market dynamics. The exploration begins by analyzing the mean sale price per year of the Ames housing market from 2006 to 2010.

Figure 1: The mean sale prices of houses in the Ames housing market from 2006 to 2010

Figure 1 showcases the mean sale prices of houses in the Ames housing market from 2006 to 2010, demonstrating an intriguing trend. The market experienced steady growth from 2006 to 2007, with the average sale price increasing from $180 879.60 to $184 728.70. This upswing can be attributed to a growing economy, increased housing demand, and possible inflationary pressures. However, in 2008, the average sale price slightly decreased to $178 587.30, which might be linked to the onset of the global financial crisis during that period, leading to a slowdown in the housing market. Despite the challenging financial crisis, the market exhibited resilience, rebounding to a mean sale price of $181 026.80 in 2009 and maintaining relative stability. In 2010, there was a more significant drop in the mean sale price to $172 323.80, which could be associated with the lingering effects of the financial crisis and the slower recovery process.

Next, the *SalePrice* variable will be explored using box plot statistics, revealing valuable insights into the distribution of house sale prices, including the median, interquartile range, and potential outliers, to gain a comprehensive understanding of the housing market in Ames, Iowa.

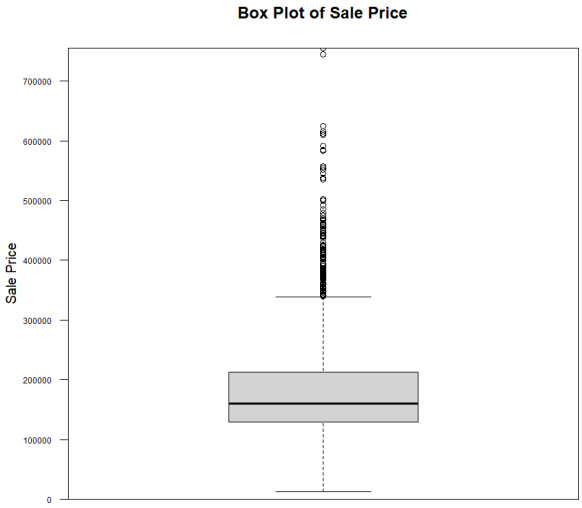

Figure 2: Box plot of the sale price

Figure 2 provides valuable insights into the distribution of house sale prices. It is apparent that the boxplot is skewed to the right. This skewness implies that the Ames housing market comprises a considerable number of moderately priced houses, while a smaller proportion of the market contains high-priced properties. The median sale price of $160 000 indicates the middle value, with half of the houses sold below this price and the other half above it. Moreover, the interquartile range (IQR) between $129 000 (1st Quartile, Q1) and $213,000 (3rd Quartile, Q3) represents the middle 50% of the data, offering an understanding of the typical range of sale prices. However, there are potential outliers, with the minimum sale price at $12 789 and the maximum at $755 000, indicating a few houses with exceptionally low or high sale prices compared to the majority of the data. To improve the predictive models, these 44 outliers will be removed, as they can disproportionately influence the model's performance and accuracy. By addressing outliers, the predictive models can be enhanced, leading to more accurate and robust predictions of house sale prices in the Ames housing market.

Finally, the issue of multicollinearity is carefully assessed by examining the correlation plot for the Ames Housing dataset. Detecting and mitigating multicollinearity is crucial for ensuring reliable interpretations and accurate predictive models. The correlation coefficients are carefully analyzed to detect highly correlated attributes that may lead to multicollinearity. Additionally, features highly correlated with the *SalePrice* variable are explored, offering valuable insights into pricing dynamics within the Ames housing market. The correlation plot serves as a vital tool for guiding further analysis and enhancing the understanding of the dataset's underlying relationships. Through this exploration, key predictors that significantly influence house sale prices are identified, paving the way for robust and insightful modeling in the study.

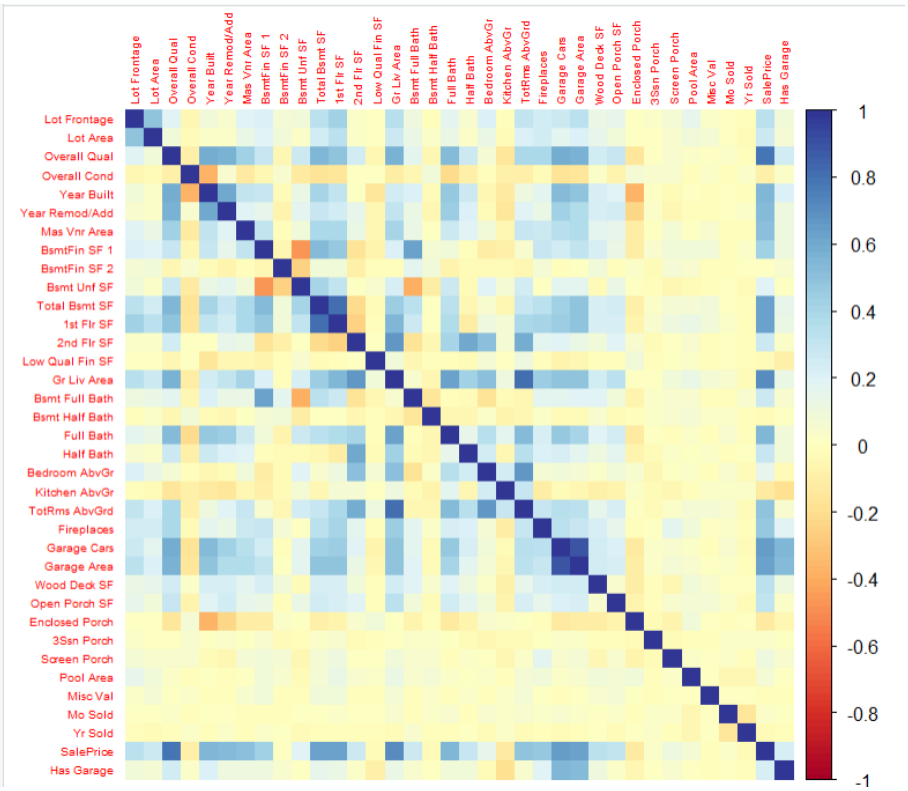

Figure 3: Correlation Plot

Figure 3 reveals crucial insights into the dataset, showing highly correlated variables with a threshold of 70%. Notably, there is a substantial positive correlation of 81.36% between *Total Bsmt SF* and *1st Flr SF*, indicating that as the total basement area increases, so does the area on the first floor. Additionally, the correlation of 80.75% between *TotRms AbvGrd* and *Gr Liv Area* suggests that houses with more rooms above ground level tend to have larger ground living areas. Concerning the target variable, *SalePrice*, it exhibits a strong correlation with two key features. First, the *Overall Qual)* shows a correlation of 79.93%, implying that higher overall quality in a house is associated with higher sale prices. Second, the *Gr Liv Area* demonstrates a correlation of 70.86%, indicating that homes with larger ground living areas tend to achieve higher selling prices. These insights are valuable for real estate decisions, aiding homeowners, professionals, and potential buyers in making informed choices about property investments.

Overall, the data exploration lays a strong foundation for understanding the Ames housing market. It provides insights into mean sale prices, identifies key predictors influencing house prices, and aids in developing accurate predictive models for informed decision-making in real estate.

3.2 Feature Selection

In the ever-expanding realm of data and machine learning, feature selection stands as a critical process that drives model performance and interpretability. As datasets grow in size and complexity, not all features (variables) contribute equally to the predictive power of a model. Some features may even introduce noise or unnecessary redundancy, leading to overfitting or decreased generalization. This section explores the important features that have been identified and recommended by the three selected techniques: Random Forest, Support Vector Regression, and XGBoost.

Through meticulous experimentation with varying numbers of important features in each model, it has been ascertained that utilizing 40 variables yields better results, establishing it as the preferred number of features. Furthermore, it is noteworthy that these three techniques yield slightly different sets of important features, contributing to a richer diversity of insights. Consequently, feature selection is done using the aforementioned techniques to ensure that the subsequent analyses are based on optimal feature subsets. In the following subsections, the paper will delve into the distinct sets of features recommended by each technique.

3.2.1 Random Forest

In the context of the random forest technique, feature importance was thoroughly assessed using two key metrics: %IncMSE and IncNodePurity. The %IncMSE metric gauges the mean decrease in accuracy or the extent to which the prediction worsens when a variable's values are randomly permuted (13). On the other hand, IncNodePurity measures the capacity of each variable to reduce node impurity when employed for splitting, thus highlighting its significance in constructing decision trees (13). After meticulous experimentation, it was observed that features selected based on the IncNodePurity metric exhibited superior performance in model accuracy. The approach demonstrated improvement in predictive capabilities and overall model efficiency, thus it was favored over utilizing the %IncMSE variables. Consequently, it was decided to proceed with the top 40 variables that were selected based on the IncNodePurity scores. Below are the top 40 variables that will be used to fit the reduced random forest model:

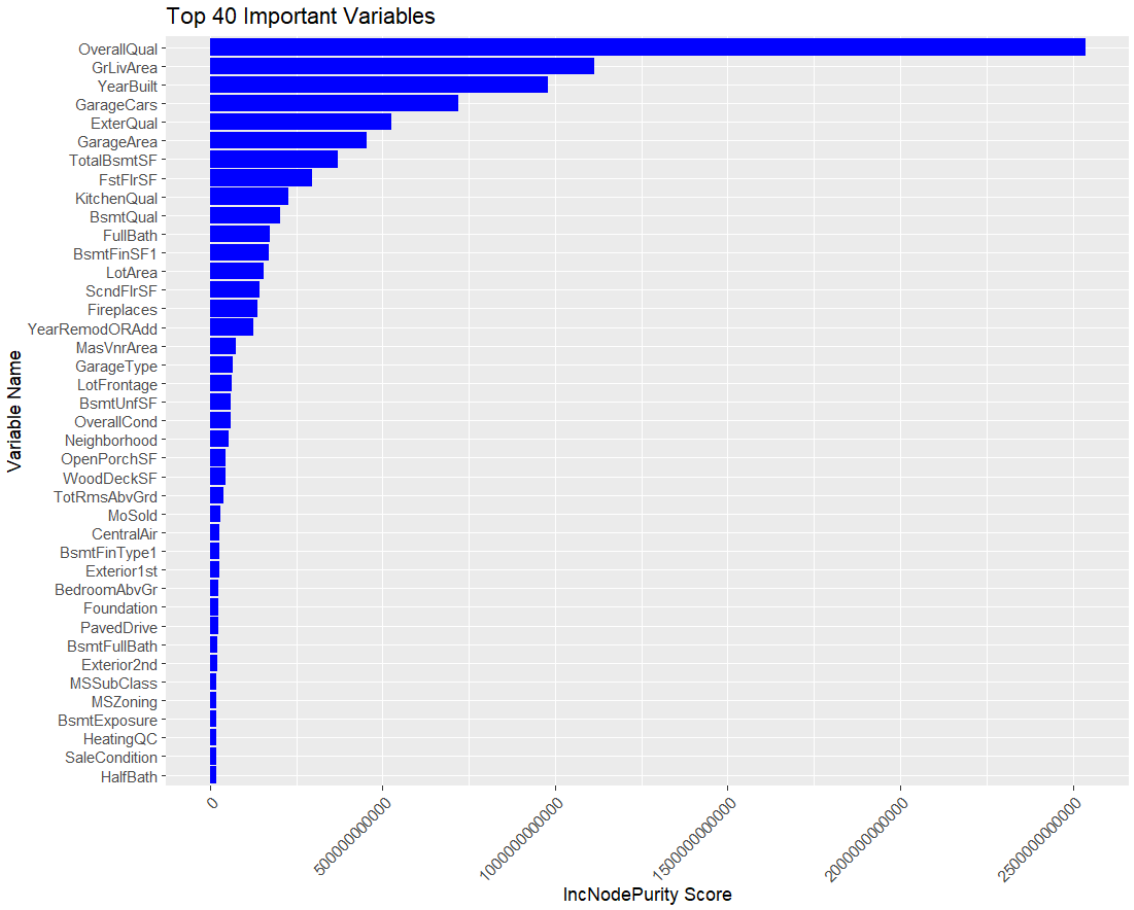

Figure 4: Random Forest Top 40 Features

Upon the examination of Figure 4, it becomes evident that certain key features wield significant influence in determining a property's sale price. Ranking these factors in order of importance, the top five influential elements are as follows: first, the overall quality of the property emerges as the most decisive determinant, indicating a positive correlation between higher overall quality and elevated sale prices. Second, the ground living area, representing the total floor space dedicated to living quarters, also exhibits substantial sway over the property's value. Third, the year built, reflecting the property's age, exerts a discernible impact, with newer constructions generally commanding higher prices. Fourth, the external quality, which assesses the condition of the property's exterior, is another crucial factor affecting the sale price. Lastly, the total basement surface area secures a spot

among the top five features, implying that properties boasting more extensive basement spaces tend to fetch higher prices.

3.2.2 Support Vector Regression

The support vector regression technique is notably lacking a dedicated R package equipped with a built-in feature selection function. In response to this deficiency, a pragmatic recourse was undertaken: the utilization of the Recursive Feature Elimination (RFE) approach. This method, available within the caret package, is used for selecting the best feature subset using a learned model, evaluating "feature importance" via a support vector machine, and eliminating the least important features (17). The primary factor guiding the selection of variables was the "overall score" metric. This metric serves as a thorough evaluation tool, combining the model's performance results across various sets of features. Through this approach, the top 40 variables that will be used to fit the reduced model are identified in Figure 5:

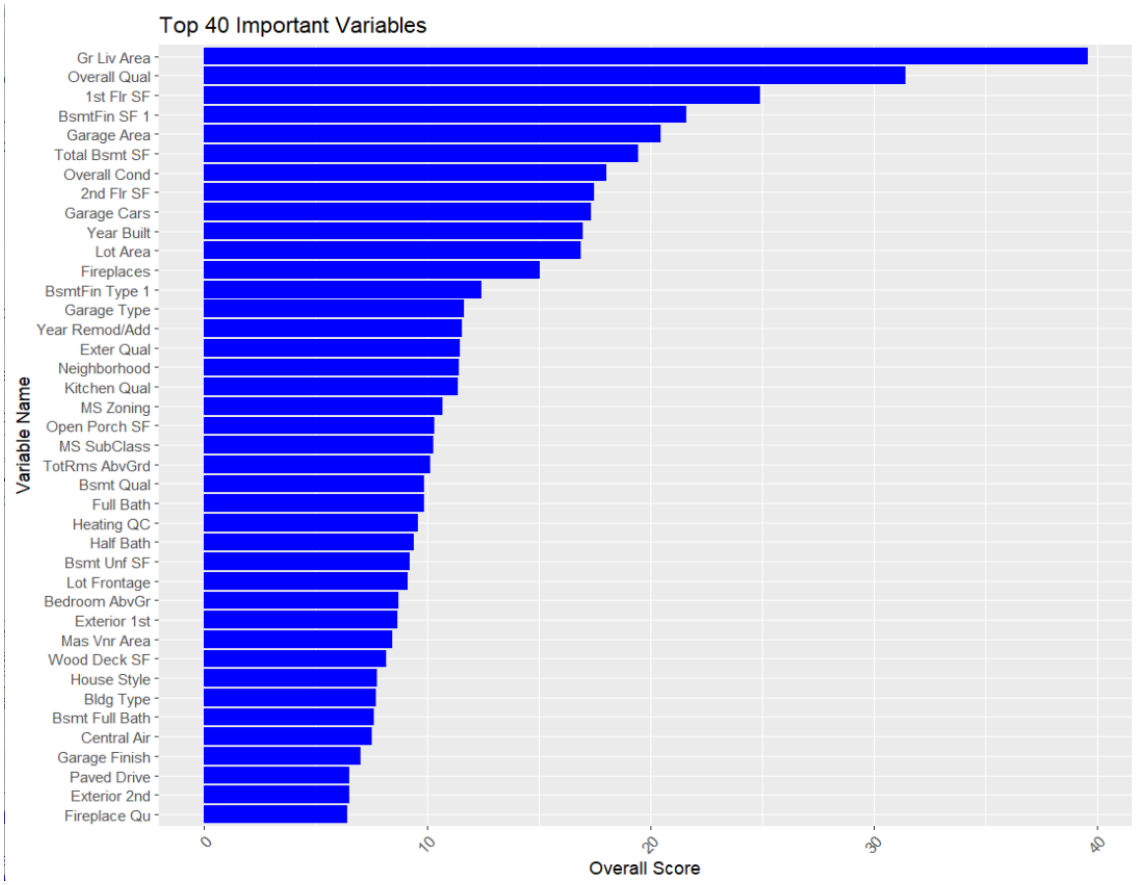

Figure 5: Support Vector Regression Top 40 Features

Five features have risen to prominence as the primary drivers of the model's predictive power. These features encompass the ground living area, the overall quality rating, the surface area of the first floor, the surface area of the primary finished section of the basement, contributing to property usability; and finally, the total basement surface area, encapsulating the extent of basement space available.

3.2.3 XGBoost

The XGBoost algorithm has been employed for feature importance assessment, utilizing three distinct metrics: gain, frequency, and cover. According to Dong et al. (9), gain represents the significance of a feature in generating predictions, with higher values indicating greater importance. Frequency denotes the relative percentage of times a specific feature occurs in the model tree, while cover refers to the relative number of observations associated with that feature. Among these three metrics, gain is considered the most relevant attribute for elucidating the relative importance of each feature. By employing the gain metric, the top 40 influential variables have been selected to fit the reduced XGBoost model. Below on Figure 6 is the visualisation of the variables.

Ranked in order of importance, the top five influential elements are as follows: first, the overall quality of the property emerges as the most decisive determinant, indicating a positive correlation between higher overall

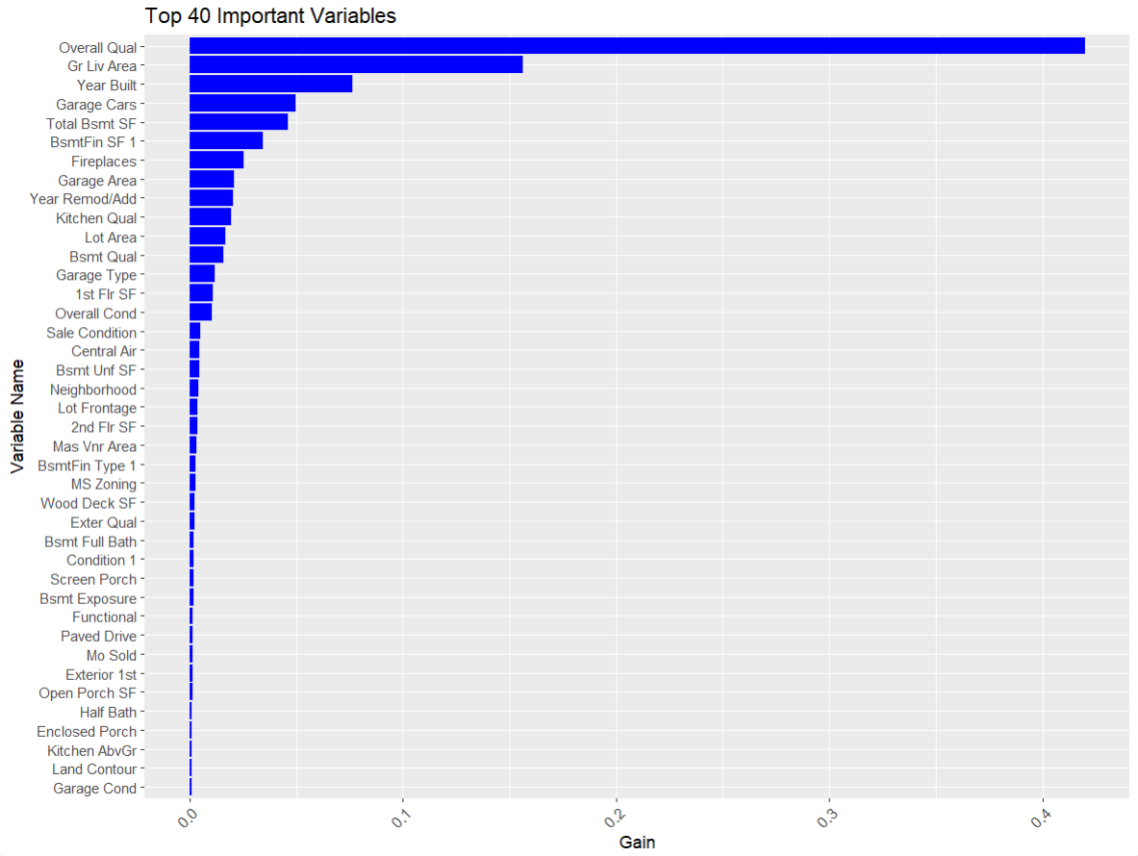

Figure 6: XGBoost Top 40 Features

quality and elevated sale prices. Second, the ground living area, representing the total floor space dedicated to living quarters, also exhibits significant sway over the property's value. Third, the year built, reflecting the property's age, exerts a discernible impact, with newer constructions generally commanding higher prices. Fourth, the number of cars a garage can accommodate is another crucial factor affecting the sale price. Lastly, the total basement surface area secures a spot among the top five features, implying that properties boasting more extensive basement spaces tend to fetch higher prices.

Across the three chosen techniques of Random Forest, Support Vector Regression, and XGBoost, a core set of features consistently emerges as key drivers of property sale prices. Notably, the overall quality of the property, ground living area, and the year built are recurrent factors that wield substantial influence in all techniques. This underscores their robust significance in predicting property values across diverse modeling approaches. While these common features offer a foundational understanding of property valuation, each technique also unveils unique insights. For instance, Random Forest emphasizes external quality and basement surface area, Support Vector Regression highlights first-floor area and primary finished basement area, while XGBoost underscores garage capacity. These distinctive features contribute nuanced perspectives, enhancing the comprehensiveness of the overall analysis and providing a more holistic grasp of the multifaceted factors at play in determining property prices.

4 Application and Results

In this section, the application and the results of the random forest, SVR and XGBoost models will be discussed. In section 4.1, the focus shifts towards the practical implementation of three distinct machine learning techniques: Random Forest, Support Vector Regression, and XGBoost. Subsequently, within section 4.2, the outcomes of the models are discussed.

4.1 Application

The three selected machine learning techniques are put into practice by building predictive models. For each technique, two distinct models are developed: a full model containing all explanatory features in the dataset

and a reduced model comprising the top 40 features selected through the respective technique's feature selection process. The main objective of the analysis is to compare these models both within and between the three selected techniques. This systematic approach allows us to evaluate the impact of feature selection on model performance and assess the effectiveness of each technique for the prediction of house prices. All data analysis and modeling building was implemented in R software using version 4.2.3 (2023-03-15 ucrt) (32). Furthermore, a seed value of 1 has been utilised to ensure reproducibility. A 70/30 train-test set split was applied into the dataset.

The random forest technique is implemented using the randomForest package. The full model is constructed with 100 trees (ntree), while the reduced model includes 50 trees, with other parameters kept at their default values. Support Vector Regression was conducted using the kernlab package. The ksvm() function was applied with the type parameter set to "eps-svr" for epsilon-support vector regression and "rbfdot" kernel to capture non-linear relationships in a higher-dimensional space. Both SVR models were built, keeping all other parameters at their default values. Lastly, XGBoost is implemented using the xgboost package in R. The specific parameters for the XGBoost model are as follows: nrounds (number of boosting rounds) is set to 1600, indicating the number of trees to build in the ensemble; objective is set to 'reg::squarederror', which represents the objective function to be optimized during training, specifically the squared error for regression tasks; eta (learning rate) is set to 0.1, controlling the contribution of each tree to the ensemble; max_depth determines the maximum depth of each decision tree in the ensemble and is set to 2; and colsample_bytree is set to 0.8, representing the fraction of features (columns) to be randomly subsampled for constructing each tree. All other parameters for the XGBoost model are kept at their default values.

Through this rigorous application, valuable insights into the strengths and weaknesses of each model within its respective technique can be gained, as well as a comprehensive comparison of the overall performance of the three techniques. The R codes and datasets can be accessed through this link: Google Drive Folder.

4.2 Results

The outcomes for each model were computed and are being assessed using metrics such as the coefficient of determination, mean squared error, root mean squared error, and mean absolute error as discussed in the Performance Evaluation section. Table 4 visualises the results observed for each model:

Table 4: Model Performance Metrics

	R^2	MSE	$RMSE$	MAE
Random Forest				
Full model	61%	1831068162	42790	29625
Reduced Model	62%	1776790450	42152	29317
Support Vector Regression				
Full model	89%	539307729	23223	15064
Reduced model	90%	466041744	21588	13945
XGBoost				
Full model	92%	413918676	20345	13115
Reduced model	92%	394419635	19860	13045

Within the Random Forest technique, a comparison between the full and reduced models reveals interesting insights. The full Random Forest model demonstrates a solid performance with an R^2 value of 61%, indicating that it explains a significant portion of the variance in the dependent variable. However, the reduced Random Forest model surpasses its full counterpart, achieving an R^2 value of 62%, suggesting a slightly better ability to capture the underlying relationships. Moreover, the reduced model outperforms the full model in terms of prediction accuracy, as evidenced by lower values for MSE, RMSE, and MAE. This implies that simplifying the model's complexity did not result in a loss of predictive power, but rather yielded more accurate predictions."

Moving to the Support Vector Regression technique, a similar trend is observed between the full and reduced models. The full Support Vector Regression model exhibits strong predictive capability with an R^2 value of 89%. However, the reduced model enhances this performance even further, achieving an R^2 value of 90%. Impressively, the reduced model consistently outperforms the full model across all performance metrics, boasting lower MSE, RMSE, and MAE values. This pattern highlights the efficiency of the reduced model in capturing the underlying patterns in the data, leading to more accurate predictions.

Within the XGBoost technique, both the full and reduced models demonstrate exceptional predictive prowess, with consistent R^2 values of 92% each. Interestingly, the reduced model maintains this high level of performance while also outperforming the full model in terms of prediction accuracy. The reduced XGBoost model yields lower

MSE, RMSE, and MAE values, signifying its ability to make predictions that are not only well-explained but also accurate.

Comparing between the techniques, the full XGBoost model emerges as the front-runner, exhibiting the highest R^2 value of 92% and the lowest prediction errors. The full Support Vector Regression model follows closely in performance, showcasing its capacity to capture complex relationships. While the full Random Forest model also delivers strong results, it appears to be slightly outperformed by the other techniques. However, the trend shifts when considering the reduced models, as the reduced Support Vector Regression model and reduced XGBoost model take the lead in terms of predictive accuracy.

While all three techniques display commendable performance, the XGBoost technique, specifically the XGBoost reduced model, seem to excel in terms of capturing patterns and generating precise predictions. Below is a visual comparison between the lowest-performing model and the highest-performing model:

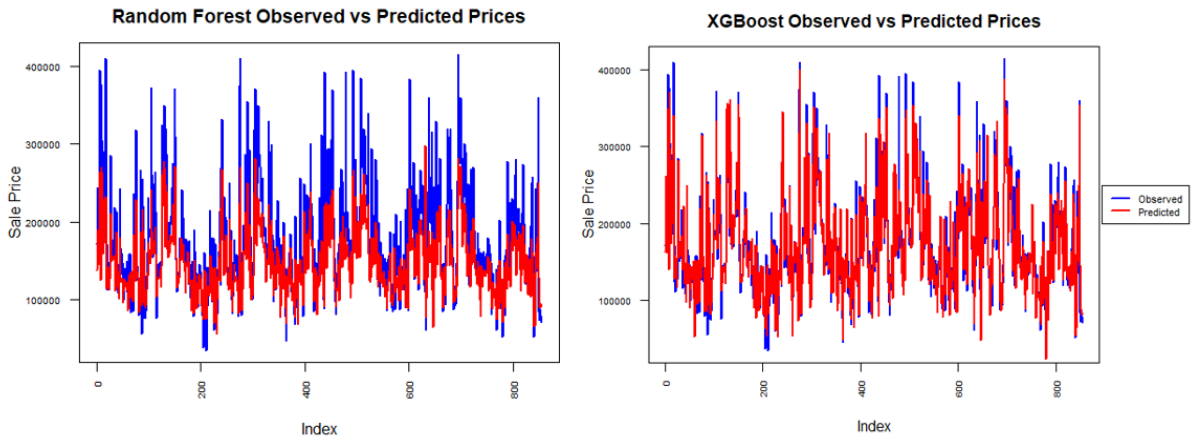

Figure 7: Visual comparison between the lowest-performing model and the highest-performing model

As depicted in Figure 7, it is evident that, in most cases, the random forest model tends to underestimate house prices. In contrast, the XGBoost model's predicted values closely align with the patterns exhibited by the observed values. While there are instances where the model slightly overestimates or underestimates, the overall performance of the XGBoost model in predicting house sale prices is commendable.

Above it all, it is important to note that the choice between these techniques should not solely rely on numerical metrics but also consider factors such as model complexity, interpretability, and other practical considerations related to the specific problem at hand.

5 Discussion

In this section, the paper discusses some of the limitations of the application and proposes potential avenues for future research that can be investigated to address these limitations.

One notable limitation lies in the dataset employed in this study, which was created in 2010. This introduces potential limitations due to the evolving dynamics of the real estate market over time. Emerging factors that have gained prominence since then, including proximity to public transportation hubs, availability of high-speed internet, and energy efficiency ratings of homes, were not included in the dataset. Integrating these contemporary features could hold the potential to enhance the predictive accuracy of the model by accounting for significant influencers of housing prices.

Another limitation is the absence of a reasonable number of domain-specific features that profoundly influence housing prices, such as local economic indicators and crime rates. The Ames Housing dataset only includes one relevant feature, which is MS Zoning. Incorporating these nuanced features could enrich the model's predictive capacity, providing a more comprehensive understanding of the intricate fluctuations in housing prices.

Hyperparameter tuning, a pivotal process for optimizing machine learning models, was not undertaken in this study due to time constraints. Implementing fine-tuned hyperparameters holds promise for further improving the model's performance, potentially yielding superior predictive outcomes.

Lastly, this study employed individual models, specifically Random Forest, Support Vector Regression (SVR), and XGBoost. Future research could explore the possibilities of combining these models through ensemble stacking. This approach capitalizes on the strengths of each model to create a more robust and accurate prediction. Additionally, considering the ensemble of these models might provide valuable insights into predictive accuracy and model stability.

6 Conclusion

This study delved into the world of house price prediction using advanced machine learning techniques. The real estate market, known for its complexity and dynamic nature driven by various factors, underscored the need for accurate and data-driven methods to estimate property values. Leveraging machine learning algorithms, the study aimed to explore the extent to which house prices could be predicted with exceptional precision and compare the performance of selected techniques.

The study placed a spotlight on three prominent supervised learning techniques: Random Forest, Support Vector Regression, and XGBoost. Through meticulous evaluation, these techniques were carefully examined to gauge their effectiveness in deciphering the intricate relationships within the housing market. The results highlighted a common thread of robust predictive capabilities across all three techniques, with XGBoost emerging as a standout performer, adept at unraveling intricate patterns and generating highly accurate predictions.

An intriguing insight arose when comparing comprehensive and simplified models within these techniques. Interestingly, simplifying model complexity often translated into enhanced prediction accuracy, shedding light on the nuanced interplay between intricacy and precision. As stakeholders consider the selection of techniques, it becomes paramount to consider more than just numerical metrics; it entails striking a harmonious balance between model intricacy and interpretability, finely aligned with the practical nuances specific to the problem domain at hand.

References

[1] M. W. Ahmad, M. Mourshed, and Y. Rezgui, *Tree-based ensemble methods for predicting pv power generation and their comparison with support vector regression*, Energy, 164 (2018), pp. 465–474.

[2] B. S. Bhati, G. Chugh, F. Al-Turjman, and N. S. Bhati, *An improved ensemble based intrusion detection technique using xgboost*, Transactions on Emerging Telecommunications Technologies, 32 (2021), p. e4076.

[3] B. Bischl, M. Lang, L. Kotthoff, J. Schiffner, J. Richter, E. Studerus, G. Casalicchio, and Z. M. Jones, *Mlr: Machine learning in r*, The Journal of Machine Learning Research, 17 (2016), pp. 5938–5942.

[4] L. Breiman, *Random forests*, Machine Learning, 45 (2001), pp. 5–32.

[5] T. Chen and C. Guestrin, *Xgboost: A scalable tree boosting system*, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[6] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho, K. Chen, R. Mitchell, I. Cano, and T. Zhou, *Xgboost: extreme gradient boosting*, R Package Version 0.4-2, 1 (2015), pp. 1–4.

[7] A. Cutler, D. R. Cutler, and J. R. Stevens, *Random forests*, Ensemble Machine Learning: Methods and Applications, (2012), pp. 157–175.

[8] D. De Cock, *Ames, iowa: Alternative to the boston housing data as an end of semester regression project*, Journal of Statistics Education, 19 (2011), pp. 1–17.

[9] J. Dong, W. Zeng, L. Wu, J. Huang, T. Gaiser, and A. K. Srivastava, *Enhancing short-term forecasting of daily precipitation using numerical weather prediction bias correcting with xgboost in different regions of china*, Engineering Applications of Artificial Intelligence, 117 (2023), p. 105579.

[10] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, *Support vector regression machines*, Advances in Neural Information Processing Systems, 9 (1996), pp. 155–161.

[11] V. Gan, V. Agarwal, and B. Kim, *Data mining analysis and predictions of real estate*, Issues in Information Systems, 16 (2015), pp. 30–36.

[12] R. Genuer, J. M. Poggi, C. Tuleau-Malot, and N. Villa-Vialaneix, *Random forests for big data*, Big Data Research, 9 (2017), pp. 28–46.

[13] C. González, J. Mira-McWilliams, and I. Juárez, *Important variable assessment and electricity price forecasting based on regression tree models: Classification and regression trees, bagging and random forests*, IET Generation, Transmission & Distribution, 9 (2015), pp. 1120–1128.

[14] W. K. Ho, B. S. Tang, and S. W. Wong, *Predicting property prices with machine learning algorithms*, Journal of Property Research, 38 (2021), pp. 48–70.

[15] T. Hothorn, K. Hornik, C. Strobl, A. Zeileis, and M. T. Hothorn, *Package 'party'*, Package Reference Manual for Party Version 0.9-998, 16 (2015), p. 37.

[16] H. Ishwaran, U. B. Kogalur, and M. U. B. Kogalur, *Package 'randomforestsrc'*, Breast, 6 (2023), pp. 1–130.

[17] H. Jeon and S. Oh, *Hybrid-recursive feature elimination for efficient feature selection*, Applied Sciences, 10 (2020), p. 3211.

[18] S. Karasu, A. Altan, S. Bekiros, and W. Ahmad, *A new forecasting model with wrapper-based feature selection approach using multi-objective optimization technique for chaotic crude oil time series*, Energy, 212 (2020), p. 118750.

[19] A. Karatzoglou, A. Smola, K. Hornik, and M. A. Karatzoglou, *Package 'kernlab'*, CRAN R Project, (2019).

[20] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, *Lightgbm: A highly efficient gradient boosting decision tree*, Advances in Neural Information Processing Systems, 30 (2017), pp. 30–52.

[21] M. Kuhn, *The caret package*, R Foundation for Statistical Computing, Vienna, Austria, (2012).

[22] M. B. Kursa and W. R. Rudnicki, *Feature selection with the boruta package*, Journal of Statistical Software, 36 (2010), pp. 1–13.

[23] M. B. Kursa, W. R. Rudnicki, and M. M. B. Kursa, *Package 'boruta'*, The R Journal, 15 (2018), p. 6615.

[24] E. LeDell, N. Gill, S. Aiello, A. Fu, A. Candel, C. Click, T. Kraljevic, T. Nykodym, P. Aboyoun, and M. Kurka, *Package 'h2o'*, DIM, 2 (2018), p. 17.

[25] A. Liaw and M. Wiener, *Classification and regression by randomforest*, R News, 2 (2002), pp. 18–22.

[26] B. Mahesh, *Machine learning algorithms - a review*, International Journal of Science and Research (IJSR), 9 (2020), pp. 381–386.

[27] A. Maurya and S. Pandey, *A study of the impact of factors affecting house pricing*, International Journal of Engineering Research and Management (IJERM), 8 (2021), pp. 1–8.

[28] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, F. Leisch, C. C. Chang, C. C. Lin, and M. D. Meyer, *Package 'e1071'*, The R Journal, (2019).

[29] A. Mutlu, A. Dogan, and A. Ozmen, *The importance of house price prediction: Evidence from machine learning models*, Journal of Real Estate Research, 41 (2019), pp. 163–197.

[30] M. Najafzadeh and S. Niazmardi, *A novel multiple-kernel support vector regression algorithm for estimation of water quality parameters*, Natural Resources Research, 30 (2021), pp. 3761–3775.

[31] P. Probst, M. N. Wright, and A. L. Boulesteix, *Hyperparameters and tuning strategies for random forest*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9 (2019), p. e1301.

[32] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2023.

[33] S. RColorBrewer and M. A. Liaw, *Package 'randomforest'*, University of California, Berkeley: Berkeley, CA, USA, (2018).

[34] A. J. Smola and B. Schölkopf, *A tutorial on support vector regression*, Statistics and Computing, 14 (2004), pp. 199–222.

[35] I. Steinwart and P. Thomann, *liquidsvm: A fast and versatile svm package*, Stat, 1050 (2017), p. 22.

[36] C. Strobl, A. L. Boulesteix, A. Zeileis, and T. Hothorn, *Bias in random forest variable importance measures: Illustrations, sources and a solution*, BMC Bioinformatics, 8 (2007), pp. 1–21.

[37] O. Uzut and S. Buyrukoglu, *Prediction of real estate prices with data mining algorithms*, Euroasia Journal of Mathematics, Engineering, Natural and Medical Sciences, 8 (2020), pp. 77–84.

[38] V. Vipin, *Impact of socio-economic factors on house pricing: A case study of kerala*, Indian Journal of Public Health Research & Development, 12 (2021), pp. 1295–1301.

[39] S. Wood and M. S. Wood, *Package 'mgcv'*, R Package Version, 1 (2015), p. 729.

[40] M. N. Wright, S. Wager, P. Probst, and M. M. N. Wright, *Package 'ranger'*, Version 0.11, 2 (2019).

[41] F. Zhang and L. J. O'Donnell, *Support vector regression*, Machine Learning: Methods and Applications to Brain Disorders, (2019), pp. 123–140.

[42] Z. Zhang, *Multiple imputation with multivariate imputation by chained equation (mice) package*, Annals of Translational Medicine, 4 (2016).

Appendix A

Table 5: Variable Description

No.	Variable	Description	Variable Type
01	PId	A unique identifier for each property. This variable is typically used for tracking and reference purposes and does not provide meaningful information about the properties themselves.	Discrete Numerical
02	MS SubClass	This variable represents the building class or type of dwelling. It includes categories such as "20" (1-Story 1946 and newer all styles), "30" (1-story 1945 and older) and others.	Categorical Nominal
03	MS Zoning	This variable represents the general zoning classification of the property with values like "RL" (Residential Low Density), "RM" (Residential Medium Density)	Categorical Nominal
04	Lot Frontage	This variable represents the size of the lot in square feet. It measures the width of the street frontage of the property in linear feet.	Continuous Numerical
05	Lot Area	The total lot size in square feet, representing the land area of the property.	Continuous Numerical
06	Street	This variable represents the type of road access to the property (e.g., "Pave" for paved road or "Grvl" for gravel road).	Categorical Nominal
07	Alley	Type of alley access with values "Grvl," "Pave," and "NA" for no alley access.	Categorical Nominal
08	Lot Shape	General shape of the property. Values include "Reg" (regular), "IR1" (slightly irregular), "IR2" (moderately irregular), and "IR3" (irregular).	Categorical Nominal
09	Land Contour	Flatness of the property with values "Lvl," "Bnk," "HLS," and "Low."	Categorical Nominal
10	Utilities	This variable represents the type of utilities available (e.g., "AllPub" for all public utilities or "NoSeWa" for no sewer and water).	Categorical Nominal
11	Lot Config	This variable describes the lot configuration, indicating whether the property is located "Inside," "Corner," "CulDSac," "FR2" (frontage on two sides), or "FR3" (frontage on three sides).	Categorical Nominal
12	Land Slope	Describes the lot configuration, indicating whether the property is located "Inside," "Corner," "CulDSac," "FR2" (frontage on two sides), or "FR3" (frontage on three sides).	Categorical Nominal
13	Neighborhood	This variable specifies the physical locations within the city of Ames, Iowa, where the properties are situated.	Categorical Nominal
14	Condition 1	This variable describes proximity to various conditions, such as "Norm" (normal), "Feedr" (feeder road), "Artery" (adjacent to arterial street), and others.	Categorical Nominal
15	Condition 2	This variable describes proximity to various conditions if more than one condition is present for a property (Similar to Condition1).	Categorical Nominal
16	Bldg Type	This variable represents the type of dwelling, such as "1Fam" (single-family), "2FmCon" (two-family conversion), "Duplex," "TwnhsE" (townhouse end unit), and "TwnhsI" (townhouse inside unit).	Categorical Nominal

Table 5: Variable Description

No.	Variable	Description	Variable Type
17	House Style	This variable describes the architectural style of the dwelling, including "1Story" (one-story house), "1.5Fin" (one and one-half story house), "1.5Unf" (one and one-half story house, unfinished), "2Story" (two-story house), "2.5Fin" (two and one-half story house, finished), "2.5Unf" (two and one-half story house, unfinished), "SFoyer" (split foyer), and "SLvl" (split level).	Categorical Nominal
18	Overall Qual	This variable represents the overall material and finish quality of the house on a scale from 1 to 10	Categorical Ordinal
19	Overall Cond	This variable represents the overall condition of the house on a scale from 1 to 10	Categorical Ordinal
20	Year Built	This variable represents the year the house was built.	Discrete Numerical
21	Year Remod/ Add	This variable represents the year when the house was remodeled or had additions made.	Discrete Numerical
22	Roof Style	This variable describes the type of roof, with options like "Gable," "Hip," "Flat," and others.	Categorical Nominal
23	Roof Matl	This variable represents the material used for the roof, including materials like "CompShg" (standard shingle), "WdShake" (wood shingles), and others.	Categorical Nominal
24	Exterior 1st	The primary exterior covering of the house, such as "VinylSd," "HdBoard," "MetalSd," and more.	Categorical Nominal
25	Exterior 2nd	Exterior covering on the house (If there is a secondary exterior covering, it is recorded here).	Categorical Nominal
26	Mas Vnr Type	This variable indicates the type of masonry veneer used on the exterior, with values like "BrkFace," "None," "Stone," and others.	Categorical Nominal
27	Mas Vnr Area	This variable represents the area of masonry veneer in square feet.	Continuous Numerical
28	Exter Qual	This variable represents the quality of the exterior material on a scale from "Ex" (Excellent) to "Po" (Poor)	Categorical Ordinal
29	Exter Cond	This variable indicates the condition of the exterior, with values from "Ex" (excellent) to "Po" (poor).	Categorical Ordinal
30	Foundation	This variable describes the type of foundation, such as "PConc" (poured concrete), "CBlock" (cinder block), "BrkTil" (brick and tile), and others.	Categorical Nominal
31	Bsmt Qual	This variable describes the type of foundation, such as "PConc" (poured concrete), "CBlock" (cinder block), "BrkTil" (brick and tile), and others.	Categorical Ordinal
32	Bsmt Cond	This variable represents the general condition of the basement, with values from "Ex" (excellent) to "NA" (no basement).	Categorical Ordinal
33	Bsmt Exposure	This variable indicates the walkout or garden level basement exposure, with values like "Gd" (good), "Av" (average), "Mn" (minimal), and "No" (no exposure).	Categorical Ordinal
34	BsmtFin Type 1	This variable describes the quality and type of the first finished basement area in residential properties that have a finished basement. It provides information about the overall finish and condition of the primary finished basement space.	Categorical Ordinal
35	BsmtFin SF 1	This variable represents the finished square footage of the first type of basement area.	Continuous Numerical

Table 5: Variable Description

No.	Variable	Description	Variable Type
36	BsmtFin Type 2	This variable provides information about the quality and type of a potential second finished basement area in residential properties that have more than one finished basement area. It serves as a complement to the "BsmtFinType1" variable	Categorical Ordinal
37	BsmtFin SF 2	This variable represents the finished square footage of the second type of basement area.	Continuous Numerical
38	Bsmt Unf SF	This variable represents the unfinished square footage of the basement.	Continuous Numerical
39	Total Bsmt SF	This variable represents the total square footage of the basement area.	Continuous Numerical
40	Heating	This variable represents the type of heating system used in the property with values like "Floor" (radiant floor heating), "GasA" (forced air heating with natural gas), "GasW" (hot water heating), "Grav" (gravity heating), "OthW" (other heating methods), "Wall" (wall-mounted heating), and "None" (no specific heating system).	Categorical Nominal
41	Heating QC	This variable indicates the heating quality and condition rating with values such as "Ex" (excellent), "Gd" (good), "TA" (typical/average), "Fa" (fair), and "Po" (poor).	Categorical Ordinal
42	Central Air	This variable indicates whether the property has central air conditioning with values "Y" (yes) and "N" (no).	Categorical Nominal
43	Electrical	This variable describes the electrical system used in the property with values like "SBrkr" (standard circuit breakers and Romex), "FuseA" (fuse box over 60 AMP and all Romex wiring), "FuseF" (60 AMP fuse box and mostly Romex wiring), "FuseP" (60 AMP fuse box and mostly knob and tube wiring), and "Mix" (mixed).	Categorical Nominal
44	1st Flr SF	This variable represents the square footage of the first floor.	Continuous Numerical
45	2nd Flr SF	This variable denotes the square footage of the second floor.	Continuous Numerical
46	Low Qual Fin SF	This variable indicates the square footage of low-quality finished space.	Continuous Numerical
47	GrLivArea	This variable represents the above-ground living area's square footage.	Continuous Numerical
48	Bsmt Full Bath	This variable represents the number of full bathrooms in the basement.	Discrete Numerical
49	Bsmt Half Bath	This variable indicates the number of half bathrooms in the basement.	Discrete Numerical
50	Full Bath	This variable represents the number of full bathrooms above grade (in the house).	Discrete Numerical
51	Half Bath	This variable denotes the number of half bathrooms above grade (in the house).	Discrete Numerical
52	Bedroom AbvGr	This variable represents the number of bedrooms above basement level.	Discrete Numerical
53	Kitchen AbvGr	This variable indicates the number of kitchens above basement level.	Discrete Numerical
54	Kitchen Qual	This variable represents the kitchen quality on a scale from "Ex" (Excellent) to "Po" (Poor)	Categorical Ordinal
55	TotRmsAbvGrd	This variable represents the total rooms above grade (excluding bathrooms).	Discrete Numerical

Table 5: Variable Description

No.	Variable	Description	Variable Type
56	Functional	This variable variable describes the home's functionality rating with values such as "Typ" (typical functionality), "Min1" (minor deductions 1), "Min2" (minor deductions 2), "Mod" (moderate deductions), "Maj1" (major deductions 1), "Maj2" (major deductions 2), "Sev" (severely damaged), and "Sal" (salvage only).	Categorical Ordinal
57	Fireplaces	This variable indicates the number of fireplaces in the house.	Discrete Numerical
58	Fireplace Qu	This variable represents the fireplace quality rating with values like "Ex" (excellent), "Gd" (good), "TA" (typical/average), "Fa" (fair), "Po" (poor), and "NA" (no fireplace).	Categorical Ordinal
59	Garage Type	This variable specifies the garage location with values such as "2Types" (more than one type of garage), "Attchd" (attached to the home), "Basment" (basement garage), "BuiltIn" (built-in garage, typically with a room above), "CarPort" (car port), "Detchd" (detached from the house), and "NA" (no garage).	Categorical Nominal
60	Garage Yr Blt	This variable represents the year the garage was built.	Discrete Numerical
61	Garage Finish	This variable describes the interior finish of the garage with values like "Fin" (finished), "RFn" (rough finished), "Unf" (unfinished), and "NA" (no garage).	Categorical Ordinal
62	Garage Cars	This variable represents the number of cars the garage can accommodate.	Discrete Numerical
63	Garage Area	This variable denotes the size of the garage in square feet.	Continuous Numerical
64	Garage Qual	This variable represents the garage quality rating with values like "Ex" (excellent), "Gd" (good), "TA" (typical/average), "Fa" (fair), "Po" (poor), and "NA" (no garage).	Categorical Ordinal
65	Garage Cond	This variable indicates the garage condition rating with values such as "Ex" (excellent), "Gd" (good), "TA" (typical/average), "Fa" (fair), "Po" (poor), and "NA" (no garage).	Categorical Ordinal
66	Paved Drive	This variable indicates whether the driveway is paved and is a categorical ordinal variable with values like "Y" (yes), "P" (partial paved), and "N" (no).	Categorical Ordinal
67	Wood Deck SF	This variable represents the square footage of wood deck area.	Continuous Numerical
68	Open Porch SF	ThIs variable denotes the square footage of open porch area.	Continuous Numerical
69	Enclosed Porch	This variable indicates the square footage of enclosed porch area.	Continuous Numerical
70	3Ssn Porch	This variable represents the square footage of three-season porch.	Continuous Numerical
71	Screen Porch	This variable denotes the square footage of screen porch area.	Continuous Numerical
72	Pool Area	This variable indicates the square footage of the pool area.	Continuous Numerical
73	Pool QC	This variable represents the pool quality rating with values like "Ex" (excellent), "Gd" (good), "TA" (typical/average), "Fa" (fair), and "NA" (no pool).	Categorical Ordinal
74	Fence	This variable describes the fence quality rating with values such as "GdPrv" (good privacy), "MnPrv" (minimum privacy), "GdWo" (good wood), "MnWw" (minimum wood/wire), and "NA" (no fence).	Categorical Ordinal

Table 5: Variable Description

No.	Variable	Description	Variable Type
75	Misc Feature	This variable represents miscellaneous features not covered by other variables and is a categorical nominal variable with values like "Elev" (elevator), "Gar2" (second garage), "Othr" (other), "Shed" (shed), and "TenC" (tennis court).	Categorical Nominal
76	Misc Val	This variable represents the value of miscellaneous features.	Continuous Numerical
77	Mo Sold	This variable denotes the month when the property was sold.	Discrete Numerical
78	Yr Sold	This variable indicates the year when the property was sold.	Discrete Numerical
79	Sale Type	This variable represents the type of sale with values such as "WD" (Warranty Deed - Conventional), "CWD" (Warranty Deed - Cash), "VWD" (Warranty Deed - VA Loan), "New" (New home), "COD" (Court Officer Deed/Estate), and others.	Categorical Nominal
80	Sale Condition	This variable describes the condition of the sale with values like "Normal" (normal sale), "Abnorml" (abnormal sale), "AdjLand" (adjoining land purchase), "Alloca" (allocation - two linked properties with separate deeds), "Family" (sale between family members), and "Partial" (home was not completed when last assessed).	Categorical Nominal
81	Sale Price	This variable is the target variable representing the sale price of the property. Its value represents the monetary amount for which the property was sold.	Continuous Numerical