U.S. House Rental Price Prediction and Classification

Karatzas Charalampos

xarhskaratzas@gmail.com

Abstract

The escalating demand for urbanization, coupled with constrained housing supply and inflationary pressures, has emerged as a primary catalyst for surging housing prices. This study presents the development of advanced Machine Learning (ML) and Deep Learning (DL) models aimed at forecasting and categorizing housing prices across the United States. Leveraging the Kaggle dataset *"Apartments for Rent Classified"* (Anan, 2024), comprising 99,492 entries, we implemented and rigorously evaluated a diverse array of regression and classification algorithms. In the regression domain, the Random Forest model demonstrated superior predictive capability, yielding an optimized coefficient of determination ($R^2 = 0.7953$) and Root Mean Square Error (RMSE) values of 201.11 (training) and 241.85 (cross-validation). For classification, the XGBoost algorithm outperformed alternative models, achieving a test accuracy of 76.53% and an F1 score of 0.7658. These findings underscore the efficacy of ensemble-based learning techniques—particularly Random Forest and XGBoost—in capturing complex housing market patterns. The study offers valuable, data-driven insights to real estate stakeholders seeking to navigate and anticipate market dynamics through predictive analytics.

Keywords: Real Estate, Machine Learning, Deep Learning, Regression, Classification, Optimization

1 Introduction

Real estate has consistently been one of the most steadily developing and expanding sectors in the market. In recent years, a growing proportion of the population has migrated to cities and urban areas, thereby increasing the demand for housing. This trend has driven significant growth in the real estate sector. In particular, the U.S. housing market is characterized by notable price fluctuations, raising concerns among all stakeholders involved—buyers, renters, and investors. Accurate real estate price prediction and classification play a crucial role in supporting informed decision-making. These capabilities benefit the aforementioned stakeholders by reducing the risk of overpayment for buyers and renters, optimizing investment strategies to maximize return on investment (ROI), mitigating potential losses, personalizing property searches, and analyzing demand and pricing trends within specific market segments. In light of these considerations, the following research question has been derived:

> *How accurately can housing prices in the U.S. be predicted, and housing price categories be classified?*

This study seeks to address these concerns by applying Machine Learning (ML) and Deep Learning (DL) techniques, alongside data preprocessing and feature selection, to enhance the accuracy of price prediction and classification. The primary objectives of this research are:

a) To identify the most influential features affecting housing prices; b) To develop, optimize, and evaluate various ML and DL algorithms for improved estimation accuracy.

2 Related Work

Numerous recent studies on housing price prediction in the United States have focused on specific states, leveraging Machine Learning (ML) and Deep Learning (DL) methods. This section highlights several notable and publicly accessible studies that contribute to this growing field of research.

Liao's research presents a comparative evaluation of multiple ML models, a comprehensive approach not previously undertaken using the same dataset. The dataset, sourced from Kaggle, contains detailed listings and pricing attributes specific to the Seattle housing market. Liao trained eight distinct ML models, including Linear Regression (Lasso), ElasticNet, Kernel Ridge Regression (KRR), Random Forest, Gradient Boosting, XGBoost regression, and two ensemble methods—AveragingModels and StackingAveragedModels. Among these, the stacking ensemble model achieved the highest performance, with an R^2 score of 0.7771 and an RMSE of 0.2328, demonstrating strong predictive

capability in a high-cost urban environment (Liao, 2024).

In another study, Sharma applied multiple ML models to a housing dataset from Ames, Iowa, evaluating the performance of XGBoost, Support Vector Regression, Random Forest Regressor, Multilayer Perceptron, and Multiple Linear Regression. The findings indicate that XGBoost outperformed the other models, achieving the lowest Mean Squared Error (MSE) of 0.001, thus confirming its effectiveness in generating precise property price estimates (Sharma et al., 2024).

Similarly focused on Ames, Iowa—but utilizing a different dataset—Yu and Wu examined both regression and classification approaches to predict housing prices. The dataset, titled the *Ames Housing Dataset*, was obtained from the Ames Assessor's Office and includes data on residential properties sold between 2006 and 2010. The study employed Lasso, Ridge, Support Vector Machine (SVM), and Random Forest (RF) regressors for regression tasks, and Logistic Regression, Naive Bayes, SVM, and Random Forest for classification. Results revealed that the best regression model was the SVR with a Gaussian kernel, achieving an RMSE of 0.5271, while for classification, the top-performing model was the SVC with a linear kernel, attaining an accuracy of 0.6913 (Yu & Wu, 2016).

Finally, Yazdani aimed to construct more accurate models using both ML and DL techniques by analyzing a housing transaction dataset from Boulder, Colorado. This dataset included features such as property size, age, location, amenities, and price. The study utilized Random Forest (RF), Artificial Neural Networks (ANN), and Hedonic Regression as a baseline. The RF model achieved the highest accuracy with an R^2 value of 0.86 and an RMSE of 0.21 on the test set. ANN followed closely with R^2/RMSE scores of 0.85/0.22, respectively. These findings underscore the ability of RF and ANN models to capture non-linear relationships between housing features and prices more effectively than the traditional Hedonic model (Yazdani, 2024).

3 Conceptual Framework

3.1 Determinants of Price and Market Complexity

The U.S. housing market is highly dynamic and influenced by a wide range of factors, including urbanization, inflation, and limited housing supply. In terms of pricing, numerous elements contribute to its complexity—most notably demand and location-based attributes such as proximity to city centers and access to transportation. Understanding the variation in housing prices and effectively capturing this complexity is essential for generating more accurate price estimations and for classifying properties into distinct pricing categories.

3.2 Data-Driven Modeling Approaches

To address these problems, we begin our analysis by visualizing the data to gain an initial understanding of its characteristics. Concurrently, we conduct data cleansing procedures, including outlier removal, followed by the handling of missing values, application of encoding techniques, and dimensionality reduction. Once the dataset has been appropriately processed, we apply supervised machine learning techniques—specifically, regression and classification. On one hand, we develop various regression algorithms to predict the actual rental price of listings based on their attributes. On the other hand, we implement classification models to categorize listings into predefined price tiers (low, medium, and high).

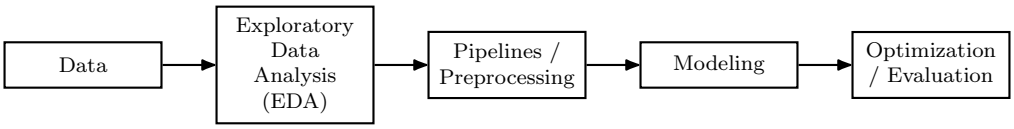

Figure 1: Conceptual Framework Diagram

These concepts will be further analyzed on the next section of methodology.

4 Methodology

4.1 Dataset Description

The dataset utilized in this research comprises 99,492 apartment rental listings, each described by 22 variables encompassing both numerical (e.g., price, bedrooms, bathrooms, latitude, longitude) and categorical (e.g., city, state, category) features. The data, sourced from an online rental platform, exhibits the typical heterogeneity of real-world datasets, including missing values, outliers, and pronounced class imbalance-most notably in the category variable, where "housing/rent/apartment" is overwhelmingly dominant. Several fields, such as address and pets_allowed, contain significant proportions of missing data, necessitating a robust and systematic approach to preprocessing and analysis. After removing duplicates and some columns that had no value for the analysis like currency type, category of listing, id, the final dataset that was introduced had a shape of 91,945 rows and 10 columns.

amenities	bathrooms	bedrooms	has_photo	pets_allowed	price	sq ft	state	latitude	longitude
NA	1.0	1.0	Thumbnail	Cats	2195	542	CA	33.8520	-118.3759
NA	1.5	3.0	Thumbnail	Cats, Dogs	1250	1500	VA	37.0867	-76.4941
NA	2.0	3.0	Thumbnail	NA	1395	1650	NC	35.8230	-78.6438
NA	1.0	2.0	Thumbnail	Cats, Dogs	1600	820	CA	38.3622	-121.9712
NA	1.0	1.0	Thumbnail	Cats, Dogs	975	624	NM	35.1038	-106.6110

Table 1: Final Dataset Head(5)

Column	Missing Count	Missing %
amenities	15,198	15.27%
bathrooms	63	0.06%
bedrooms	117	0.12%
has_photo	0	0.00%
pets_allowed	55,609	55.89%
price	0	0.00%
square_feet	0	0.00%
state	271	0.27%
latitude	25	0.03%
longitude	25	0.03%

Table 2: Missing Values in the Initial Dataset Before Filtering and Preprocessing

4.2 Data Analysis Process

The data analysis process followed some key steps; Firstly, an exploratory data analysis was performed to inspect the data. Views of the first few rows, check of the data types, and identification of missing values. The value counts for categorical variables were analyzed to assess the class imbalance and data quality. Secondly, visualization techniques were applied for both univariate and multivariate analysis, such as histograms, boxplots, geographical plots (see Fig. 2, 4 - 18), and correlation matrices, using Matplotlib and Seaborn to understand distributions, detect outliers, and explore relationships between variables. An automated profiling was also performed for a comprehensive overview of the statistics and anomalies of the data set. All of those were very important steps to finalize the pre-processing pipelines, proceed to train, and finally evaluate the regression and classification models.

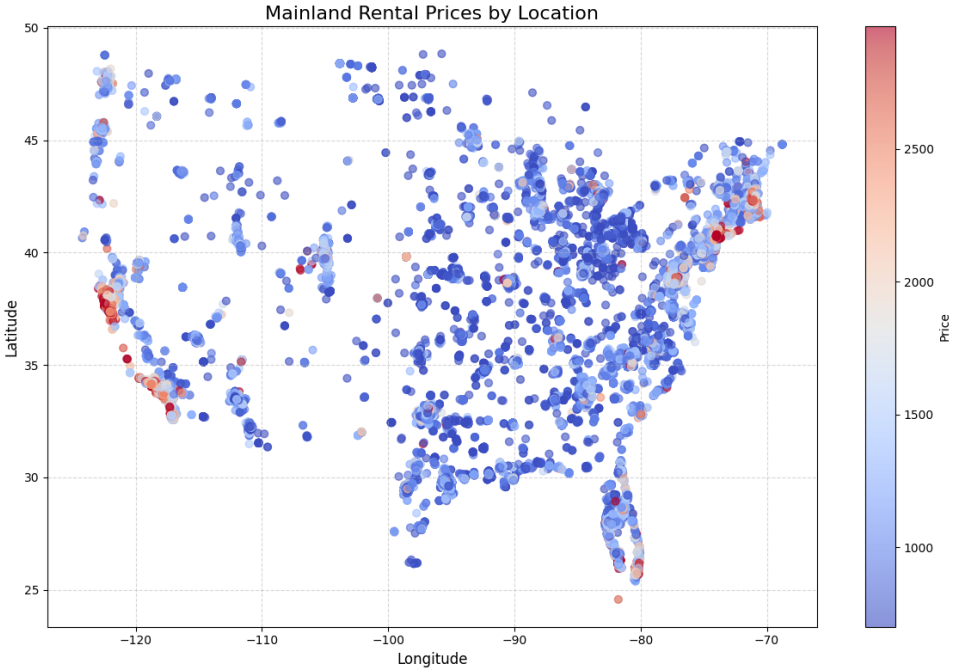

Figure 2: Rental price heatmap - density of listings

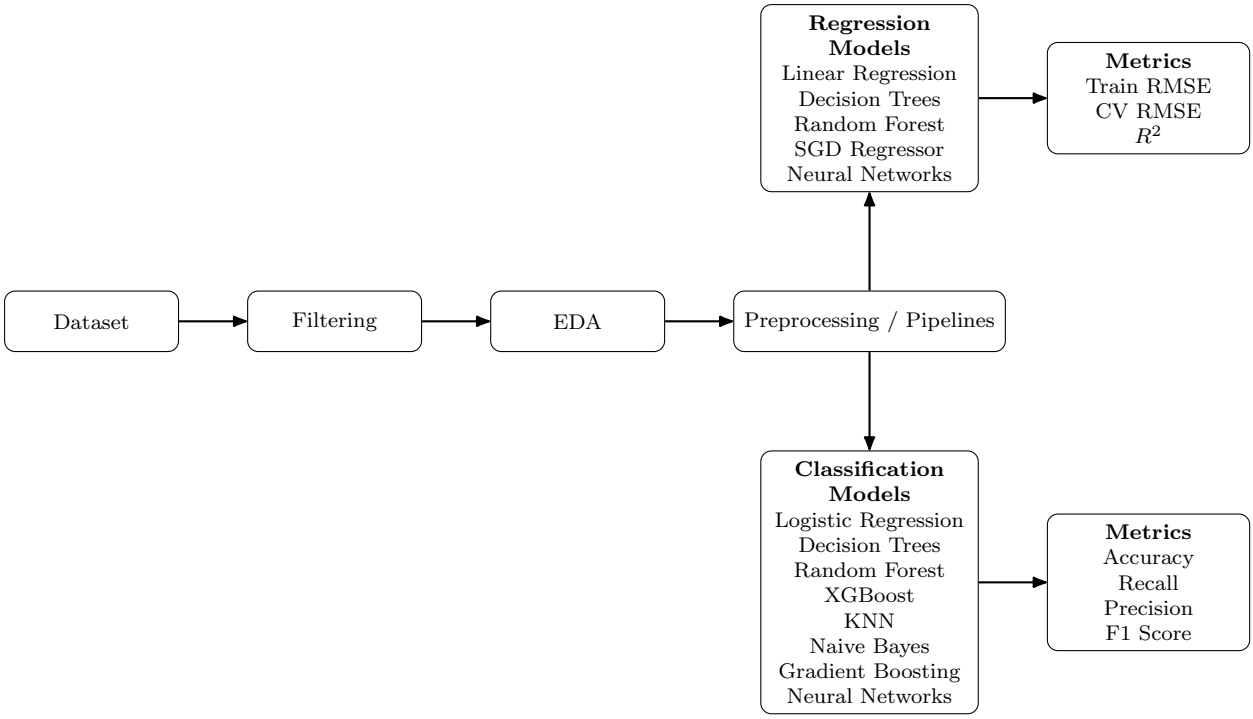

Figure 3: Data Analysis Process Flowchart

4.2.1 Data Filtering, Transformation, and Combination

Following an initial exploration of the dataset, a systematic data filtering methodology was applied. U.S. states with fewer than 100 listings were consolidated into a single "Other" category to reduce dimensionality and mitigate sparsity in the state feature. Duplicated records were removed, and outliers were identified and eliminated using the Interquartile Range (IQR) method to improve data quality. Listings with irrelevant or rare categories (e.g., categories with very few to minimal listings) were also dropped to reduce noise and dimensionality.

Missing values in the amenities and pets_allowed columns, which exhibited a high proportion of missing entries, were imputed with placeholder values such as "Not available" and "None", respectively. This approach was adopted to retain potentially useful categorical information while avoiding the risks associated with inappropriate statistical imputation. These imputations were conducted prior to addressing missing values in other variables, where more conventional techniques (e.g., median imputation) were subsequently employed.

Feature engineering techniques were then applied to enhance the dataset. Latitude and longitude coordinates were clustered with K-means algorithm to reveal potential geographic rental patterns. Additionally, a new categorical feature price_category was created by segmenting the continuous price variable into three ordinal classes: "low", "average", and "high". This derived feature was later utilized for classification tasks in the modeling phase.

4.2.2 Data Preprocessing

Data preprocessing was a critical phase in preparing the dataset for analysis. A custom transformer class was implemented to process the amenities column, which contains free-text data, by parsing and applying one-hot encoding. For numerical features—such as bathrooms and bedrooms—a dedicated preprocessing pipeline was constructed using scikit-learn's SimpleImputer to handle missing values (using the median) and StandardScaler to normalize the distributions.

Categorical features, excluding amenities, were transformed using OneHotEncoder. Both the numerical and categorical pipelines, along with the custom amenities transformer, were orchestrated through scikit-learn's ColumnTransformer, ensuring modular and organized feature processing. These components were integrated into a single Pipeline object to promote reproducibility and streamline the workflow.

Additionally, the dataset was partitioned into training and testing subsets using train_test_split, enabling unbiased model evaluation and validation during subsequent stages of the analysis. Crucially, the pipeline was fitted exclusively on the training data after the dataset was partitioned into training and test sets. The fitted pipeline was then used to transform both the training and test data, ensuring that no information from the test set influenced the learning of preprocessing parameters. This approach is essential for preventing data leakage and obtaining unbiased estimates of model performance (Géron, 2019).

To address the high dimensionality resulting from one-hot encoding of categorical variables and to mitigate the curse of dimensionality, Random Forest Feature Importance and Principal Component Analysis (PCA) was applied. The number of principal components was chosen to retain a high proportion of the total variance (typically 95% or higher), thereby preserving most of the dataset's informational content while reducing computational complexity. In paraller, a Random Forest model was trained on the preprocessed data to compute feature importances. This model-agnostic feature selection method ranked features according to their contribution to the predictive-task, as measured by the mean decrease in impurity (Gini importance). The features with the highest importnace scores were selected for further modeling. This approach not only reduced dimensionality by 80 features to 40 but also enhanced model interpretability by highlighting the most influential predictors in the dataset. The combined use of these techniques resulted in a more parsimonious feature set, imporved generalization, and reduced risk of overfitting.

4.3 Data Analytics: Modeling, Methods and Tools

The analytical phase of this project encompassed both regression and classification tasks, employing a structured methodology that began with the development of baseline models

and progressed to more sophisticated algorithms through systematic optimization and regularization. This approach aligns with best practices in applied machine learning (Géron, 2019), ensuring both interpretability and rigor in model selection and evaluation.

For both regression and classification problems, initial modeling efforts focused on fitting a suite of baseline models. In regression, these models, included Linear Regression, Decision Tree Regressor, Random Forest Regressor and SGD Regressor. For classification, baseline models comprised Logistic Regression, Decision Tree Classifier, XGBClassifier, KNN, Naive Bayes, and Gradient Boosting. Each baseline model was evaluated for predictive or classifying performance and susceptibility to overfitting. Overfitting was assessed by comparing performance metrics on the training set with those obtained via cross-validation, as recommended by (Géron, 2019).

Upon selection of best-performing baseline models with as less overfitting anomalies as possible, extensive hyperparameter tuning was conducted to further enhance predictive and control model complexity. For scikit-learn models, hyperparameter optimization was performed using RandomizedSearchCV rather than the traditional GridSearchCV. This choice was motivated by computational efficiency, while GridSearchCV exhaustively evaluates all parameter combinations, RandomizedSearchCV samples a fixed number of parameter settings from specified distributions, significantly reducing computational time without sacrificing the quality of results (Géron, 2019, Ch.2). This approach is particulary advantageous for models with large hyperparameter spaces, such as Random Forests and Decision Trees (Bergstra, J., & Bengio, Y., 2012).

Regularization techniques were systematically applied to mitigate overfitting. In the case of Random Forest regularization was achieved by constraining tree depth, minimum samples per leaf, and the number of estimators-parameters known to control model complexity and reduce variance (Breiman, 2001). For XGBoost models, regularization parameters such as lambda and alpha were tuned. Notably, for tree-based models, classic regularization techniques such as dropout or L2 penalties are not directly applicable; instead, structural hyperparameters serve this role (Géron, 2019, Ch.7).

Neural networks were implemented for both regression and classification tasks using Keras'Sequential, following the methodology outlined in Geron (2019, Ch.10-11). The training set was further partioned intro training and validation subsets, a necessary step for monitoring generalization performance during neural network training. The initial architecture consisted of a baseline Multi-Layer Perceptron (MLP), whose performance was evaluated before proceeding to optimization.

Optimization of neural networks involved grid search over architectural and training hyperparameters, including the number of layers, number of neurons per layer, activation functions, optimizer types, batch sizes, and learning rates. Due to compatibility issues with the latest versions of TensorFlow and Keras, the KerasRegressor wrapper was not employed; instead, custom training loops and callbacks were used for hyperpa-

rameter exploration. Regularization was implemented through dropout layers and L2 weight penalties, both of which are established techniques for improving neural network generalization (Géron,2019, Ch. 11). Early stopping was also used to halt training when validation performance ceased to improve, further guarding against overfitting.

Model performance was rigorously evaluated using appropriate metrics for each task. In regression, the primary metrics were Root Mean Squared Error (RMSE) and the R^2, both on training set and via cross-validation. For classification tasks, performance was assessed using accuracy, precision, recall and F1-score, with additional validation metics computed to detect overfitting.

At the conclusion of each modeling task, comprehensive comparisons were made between baseline and optimized models. These comparisons included tabulated summaries of key metrics, facilitating transparent assessment of the impact of hyperparameter tuning and regularization.

4.4 Model Time Complexity Analysis

A nuanced understanding of model time complexity is essential for selecting algorithms that are not only accurate but also computationally feasible, particularly as datasets and feature spaces expand (Bergstra & Bengio, 2012; Géron, 2019; Goodfellow et al., 2016). In this study, we evaluated the computational demands of linear models, tree-based ensembles, support vector machines, and neural networks, analyzing both theoretical complexity and empirical runtime.

Linear models, such as linear and logistic regression, are highly efficient, with time complexity of $O(nm)$, where n is the number of samples and m the number of features (Géron, 2019). However, their simplicity limits their capacity to model nonlinear relationships. Tree-based ensemble methods, such as Random Forests and Gradient Boosted Trees, increase complexity to $O(knm \log n)$ due to the number of trees k and predictors (Breiman, 2001). Neural networks, particularly deep architectures, are among the most computationally intensive, with training complexity often approximated by $O(lnm)$, where l is the number of layers (Goodfellow et al., 2016).

Empirical results reflected these theoretical patterns. Linear models trained in seconds, while decision trees, Random Forests, and XGBoost required significantly more time, especially when hyperparameters were tuned. Neural networks, implemented via Keras, had the longest training durations, particularly with multiple epochs and regularization (Géron, 2019; Goodfellow et al., 2016).

Hyperparameter optimization using randomized search added further overhead. While more efficient than grid search, it still demands multiple full training cycles—particularly costly for deep or ensemble models (Bergstra & Bengio, 2012).

All scikit-learn models were trained locally on an Apple MacBook with an M2 CPU.

In contrast, neural networks and XGBoost were trained in a Google Colab environment with an NVIDIA A100 GPU, yielding training speedups of up to 50× over CPU-based execution (Rasmussen, 2023). This distinction underscores the need for GPU acceleration in deep learning, where CPU-based training is often insufficient (Goodfellow et al., 2016).

In summary, this analysis confirms the classic trade-off between computational cost and predictive accuracy: more complex models, such as Random Forests and neural networks, achieved higher accuracy but at greater training time. Model selection, therefore, must balance predictive performance with computational constraints, especially in large-scale or time-sensitive applications (Géron, 2019; Goodfellow et al., 2016).

5 Results

The analysis evaluated multiple Machine Learning and Deep Learning models for both regression and classification tasks. For regression, the Random Forest Regressor demonstrated the highest performance, as shown in Table 3, effectively reducing overfitting while achieving a strong R^2 score.

Model Type	**Model Version**	**Train RMSE**	**CV/Validation RMSE**	**Test R^2 Score**
Random Forest	Base Model	81.25	210.91	0.8442
Random Forest	Final Model (Tuned)	201.11	241.85	0.7953
Neural Network	Base Model	297.16	310.20	0.6558
Neural Network	Final Model (Tuned)	261.43	295.78	0.6852

Table 3: Regression Results

For the classification task, the XGBoost Classifier outperformed all other models, achieving a test accuracy of 76.53%, as shown in Table 4.

Regarding the classification error analysis, the confusion matrix (see Fig.29) revealed strong performance in identifying listings within the average and high price categories. However, the model exhibited lower accuracy in correctly classifying listings within the low-price category.

Model Type	**Model Version**	**Train Accuracy**	**Test Accuracy**	**Train F1**	**Test F1**	**Accuracy Gap**
Neural Network	Base Model	0.7175	0.6943	NA	0.6814	0.0232
Neural Network	Final Model	0.7498	0.7161	NA	0.7137	0.0337
XGBoost	Base Model	0.8304	0.7870	0.8307	0.7872	0.0435
XGBoost	Final Model	0.7879	0.7653	0.7882	0.7658	0.0226
Random Forest	Base Model	0.9882	0.8115	0.9882	0.8113	0.1768
Random Forest	Final Model	0.7543	0.7286	0.7525	0.7266	0.0257

Table 4: Classification Results

Feature importance analysis identified location, square footage, and the number of bedrooms and bathrooms as the most influential predictors for both regression and classification tasks (see Fig.20). It is also worth noting that, although Neural Networks are known for their ability to model complex non-linear relationships, they require large volumes of data to train effectively. This limitation partly explains why simpler ensemble models outperformed Neural Networks in this study.

5.1 Meaningful Facts

The superior performance of ensemble models, particularly Random Forest and XGBoost, underscores the existence of complex, non-linear relationships among the features influencing rental prices. The high R^2 value achieved by the Random Forest model indicates that nearly 80% of the variance in rental prices can be attributed to the selected features, highlighting the robustness and explanatory power of the model. The most influential predictors—property size, number of bedrooms and bathrooms, and geographical location—are consistent with established real estate valuation principles, where spatial and physical attributes serve as primary determinants of rental value. The relatively strong performance of the classification models further confirms that rental listings can be effectively categorized into discrete price tiers based on these features. Moreover, the role of data preprocessing and feature engineering proved to be critical in enhancing model performance. Effective feature selection, dimensionality reduction, and careful handling of missing values and outliers underscore the importance of rigorous data preparation in practical machine learning workflows. These steps are foundational to achieving reliable and accurate predictive outcomes in real-world applications.

5.2 Actionable Insights

The findings suggest several practical applications. Property managers and landlords can leverage the regression model to set competitive rental prices that reflect the true market value of their properties, particularly by emphasizing the most impactful features in their listings. For example, properties with larger square footage, more bedrooms, or more touristy locations can place higher rents. The classification model offers renters a tool to identify fair market deals and avoid overpaying for properties misaligned with their price tier. Real estate platforms can integrate these models to enhance personalized recommendations and improve user satisfaction.

5.3 Valuable Outcomes

The results have significant applicability across multiple stakeholders groups. Real estate agencies and listing platforms can deploy the regression model as an automated pricing

tool, ensuring data-driven and transparent pricing strategies. The classification model can be used to build dashboards that monitor the market segmentation and rental trends, aiding in strategic planning for developers and investors. By identifying the features most strongly associated with higher rental prices, urban planners and investors can prioritize development in high-value geographic clusters and optimize property attributes to maximize returns. Furthermore, the models provide foundation for ongoing market analysis, enabling continuous adaption to evolving rental market dynamics through regular retraining and data enrichment. In summary, the application of ensemble machine learning models to U.S. rental data yields robust, interpretable, and actionable results. These models can be directly utilized to inform pricing, investment, and policy decisions in the real estate sector, enhancing both operational efficiency and market transparency.

6 Discussion

Through the effective application of various ML and DL models for both regression and classification tasks, and based on the performance metrics analyzed in Section 5, we are confident in asserting that the models achieved solid accuracy—especially considering the inherent complexity and noise of the housing dataset. Among the evaluated models, Random Forest demonstrated superior performance in the regression task, while XGBoost outperformed others in classification. Although Neural Networks have the potential to model complex non-linear relationships, traditional ML models proved to be more suitable for medium-sized datasets such as the one used in this study.

Overall, the findings support our initial research objective: to assess whether modern AI techniques can accurately predict and classify housing prices.

This study builds upon the methodologies presented in *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* by Géron (2019), applying them specifically to the U.S. housing market. Our results are consistent with existing literature, reinforcing the observation that ensemble learning techniques are often more efficient and effective when applied to real estate datasets.

6.1 Limitations

At this point, it is important to discuss the limitations associated with the dataset. One significant issue was the high proportion of missing values in certain categories, such as amenities. These could not be dropped outright due to their substantial presence, nor could they be imputed with actual category values without introducing considerable assumptions—and potential bias. Random missingness is generally non-informative and may introduce additional noise into the dataset.

Another notable limitation is the absence of several key features that are highly rel-

evant to price estimation. Although the dataset included textual fields such as the title and description, which often contained valuable location-based information (e.g., proximity to city centers, public transportation access, quiet neighborhoods), these were not utilized due to the requirement for natural language processing techniques, which were outside the scope of this study. Additionally, the dataset lacked consistently impactful features such as property age, condition, and other structural characteristics.

In terms of computational complexity, both ensemble models and neural networks are resource-intensive, requiring considerable time for training and hyperparameter tuning—whether executed on CPU or GPU. Due to limitations in computational resources and project time constraints, we were unable to conduct extensive model fine-tuning or explore a broader set of algorithms. The Scikit-learn models employed in this study only support CPU-based training, whereas the neural network and XGBoost models were trained using GPU acceleration, as outlined in Section 4.4. While GPU usage enabled the training of more computationally demanding models, it still required several hours per iteration, limiting our ability to experiment with a wider range of hyperparameter combinations that might have further improved performance.

Finally, it is worth noting that Neural Networks are typically top-performing models when applied to large-scale datasets with high dimensionality and complexity. However, in our case, the neural networks underperformed due to the medium-sized nature of the dataset, which restricted their capacity to generalize effectively.

6.2 Ethical Considerations

Beyond the technical and data-related limitations, it is important to consider the ethical implications associated with the application of ML and DL models in the housing market. One of the most pressing ethical concerns relates to transparency, particularly in terms of bias and discrimination. Although the dataset used in this study does not contain sensitive personal information, potential geographic bias must still be acknowledged. For example, if certain regions historically feature lower-valued properties due to being home to marginalized communities, predictive models may unintentionally perpetuate this inequality by systematically undervaluing properties in those areas. Such outcomes risk reinforcing existing societal disparities through algorithmic decision-making.

Another major ethical concern involves the transparency and interpretability of the models themselves. Algorithms such as Random Forests and Neural Networks are frequently categorized as *black-box* models (SEON, n.d.). The term "black box" refers to models that, once trained, provide limited insight into their internal decision-making processes due to their complexity—such as the large number of trees and branches in ensemble methods or the layered architecture in neural networks. This opacity poses challenges in understanding how specific predictions are made, making it difficult to en-

sure accountability or to detect embedded biases.

Ethical deployment of such models requires caution, including considerations for fairness, transparency, and the potential societal impact of automated valuation systems in real estate.

7 Conclusion

This study was conducted to investigate the effectiveness of Machine Learning (ML) and Deep Learning (DL) models in predicting housing prices and classifying properties into price categories. Utilizing a dataset containing nearly 100,000 records, we implemented a comprehensive end-to-end pipeline that encompassed data preprocessing, feature engineering, dimensionality reduction, and model optimization.

Our results demonstrate that advanced models, such as Random Forest and Neural Networks, are capable of capturing complex relationships between property characteristics and market pricing. The performance metrics achieved by these models indicate satisfactory levels of accuracy and robustness, particularly in the context of a real-world housing dataset with inherent noise and complexity.

Overall, this paper integrates technical implementation with thoughtful evaluation and ethical considerations. It contributes a holistic ML/DL framework that not only advances predictive accuracy in real estate analytics but also acknowledges the social and ethical implications of automated decision-making in this domain.

8 Future Work

First and foremost, due to resource and time constraints that limited extensive model fine-tuning, future work could involve more comprehensive experimentation with hyperparameter optimization and the exploration of additional models to further improve accuracy and reduce the train-test RMSE gap.

Another promising direction for future development involves deploying, monitoring, and maintaining the trained models. This could be achieved by saving the trained model—including the complete preprocessing pipeline—in a cloud-based environment. Such a deployment would allow the model to be continuously updated and evaluated using new housing datasets. This approach would enable ongoing assessment of model performance on unseen, real-world data and provide insights into whether the model behaves equitably across different geographical regions or demographic groups.

Lastly, further enhancements in feature engineering could contribute significantly to model performance. For instance, generating new variables from existing features—such as price per square foot, total number of rooms, or average property price by state—may

provide additional predictive power. Moreover, the three clusters previously generated could serve as valuable categorical predictors, potentially offering a more interpretable and compact representation than raw geographic coordinates like longitude and latitude.

References

Anan, S. (2024). *Apartments_for_rent_classified_100k*. https://www.kaggle.com/code/saurav9786/rent-price-recommender/input

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (2nd). O'Reilly Media.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Liao. (2024). Comparative evaluation of multiple machine learning models for seattle housing price prediction. *Proceedings of the Asian Conference on Economics*, *5*, 330–336. https://doi.org/10.62051/7gzntt12

Rasmussen, C. E. (2023). Hardware acceleration for deep learning: A practical guide [arXiv preprint arXiv:2301.12345].

SEON. (n.d.). *Black box machine learning*. https://seon.io/resources/dictionary/blackbox-machine-learning/

Sharma et al. (2024). Application of multiple machine learning models to ames, iowa housing dataset. *Journal of Real Estate Analytics*.

Yazdani. (2024). Machine learning and deep learning approaches for housing price prediction in boulder, colorado. *Journal of Property Research*.

Yu, H., & Wu, J. (2016). Regression and classification approaches for predicting housing prices: A case study of ames, iowa. *International Journal of Housing Markets and Analysis*.

Appendix

A Exploratory Data Analysis (EDA)

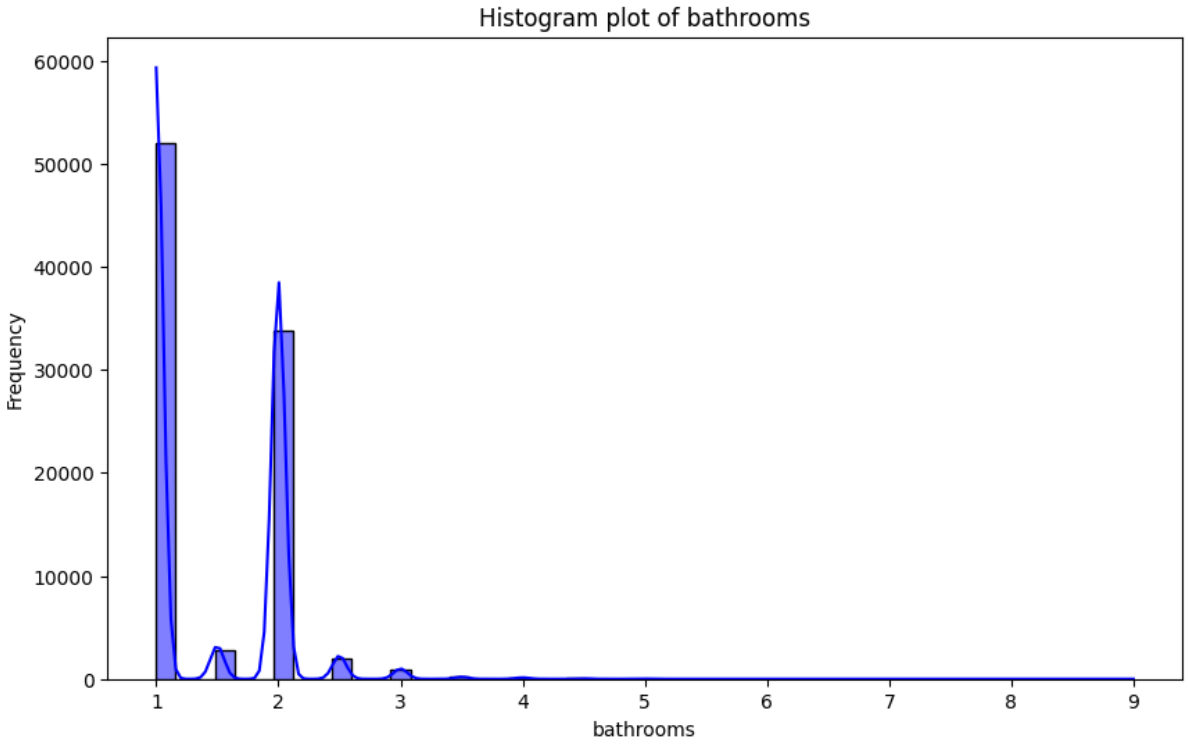

Figure 4: Histogram of bathrooms

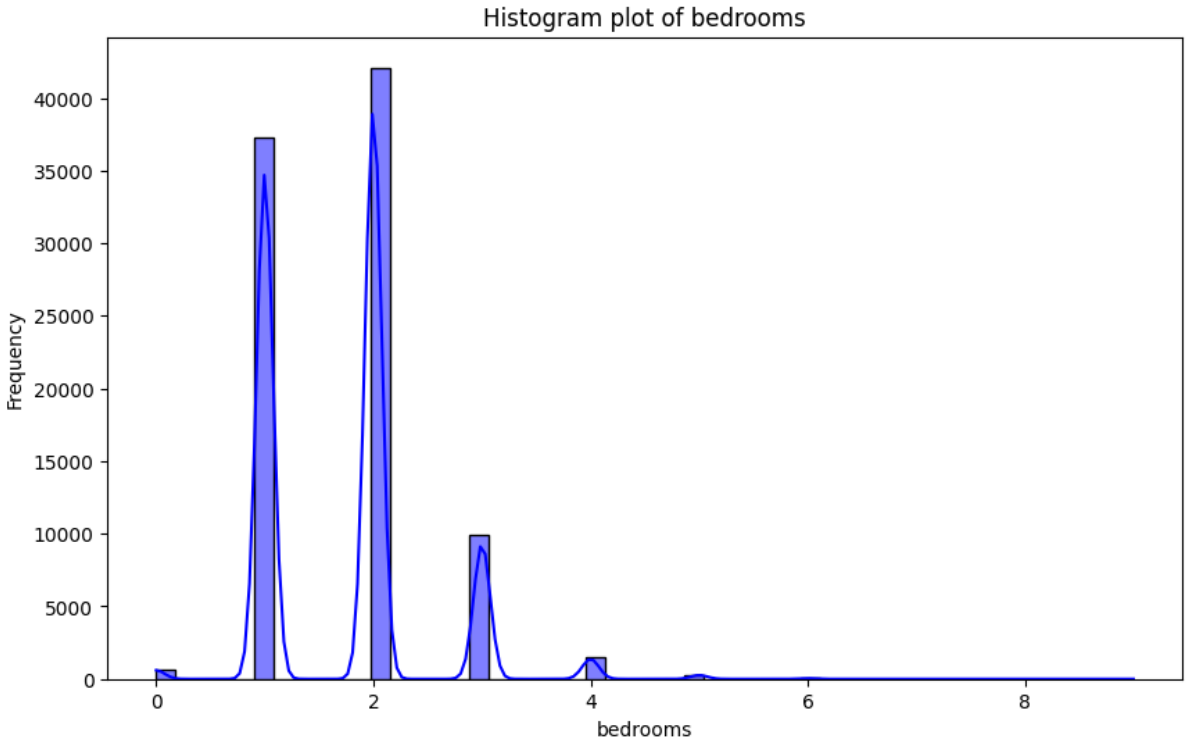

Figure 5: Histogram of bedrooms

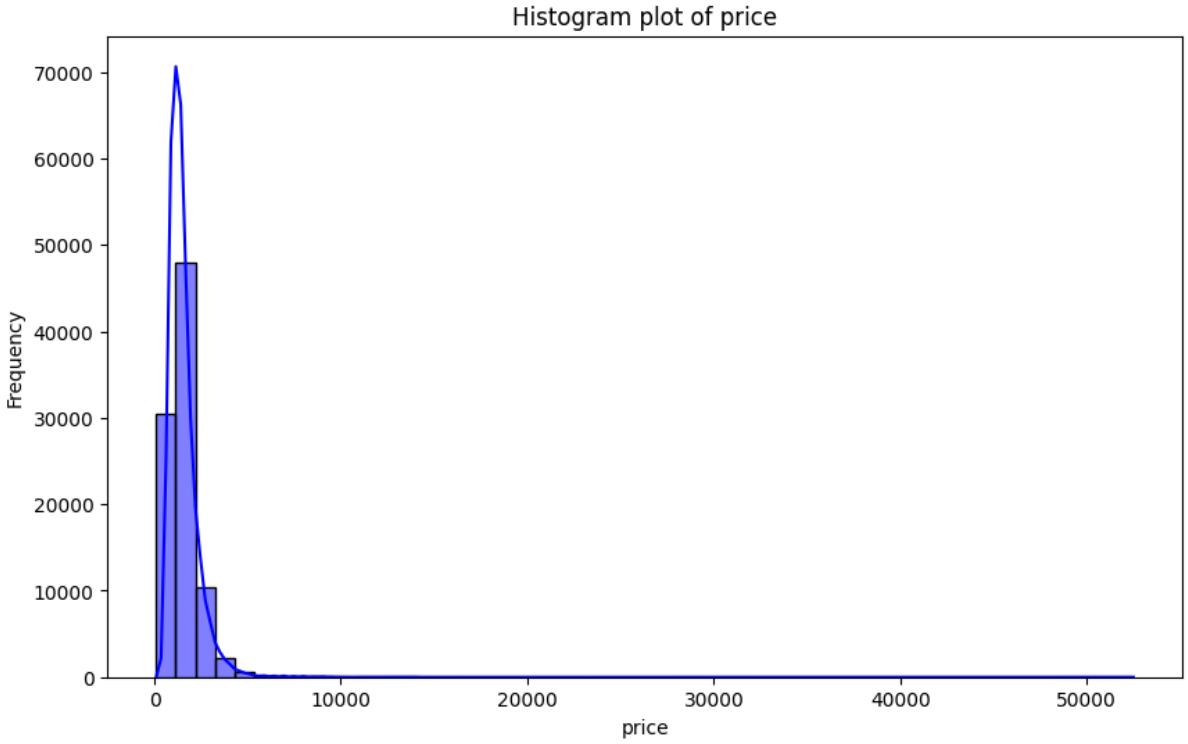

Figure 6: Histogram of price

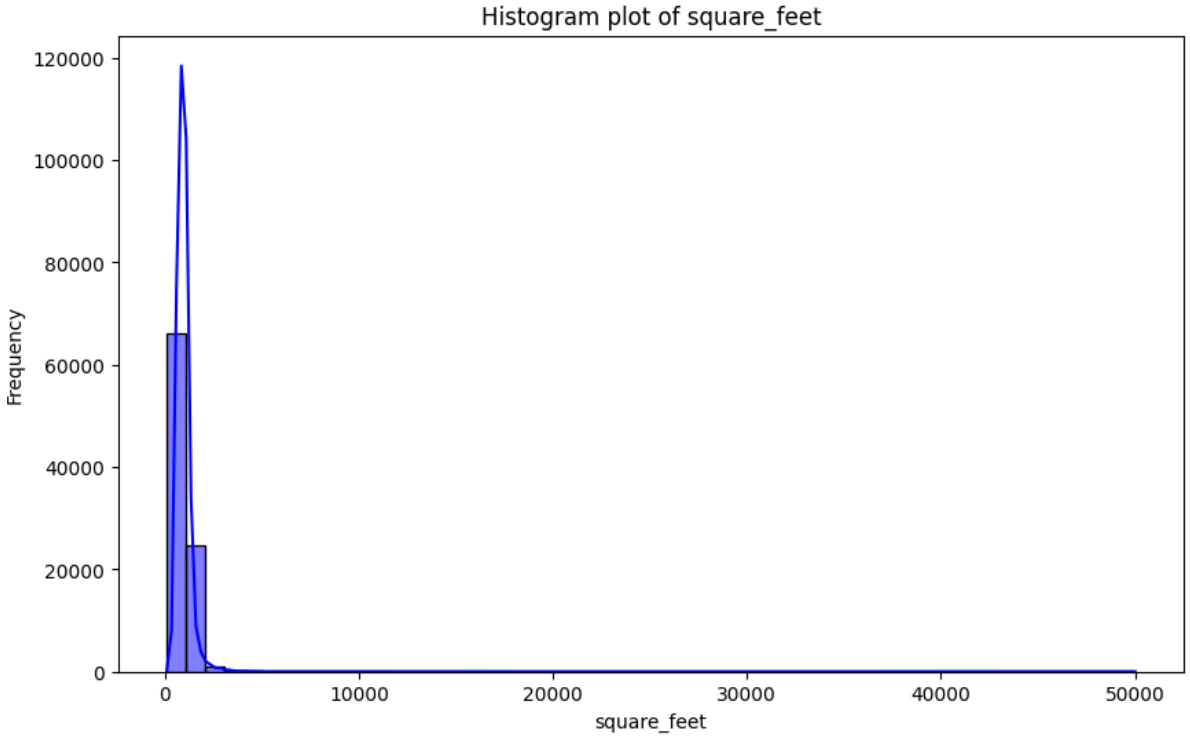

Figure 7: Histogram of square feet

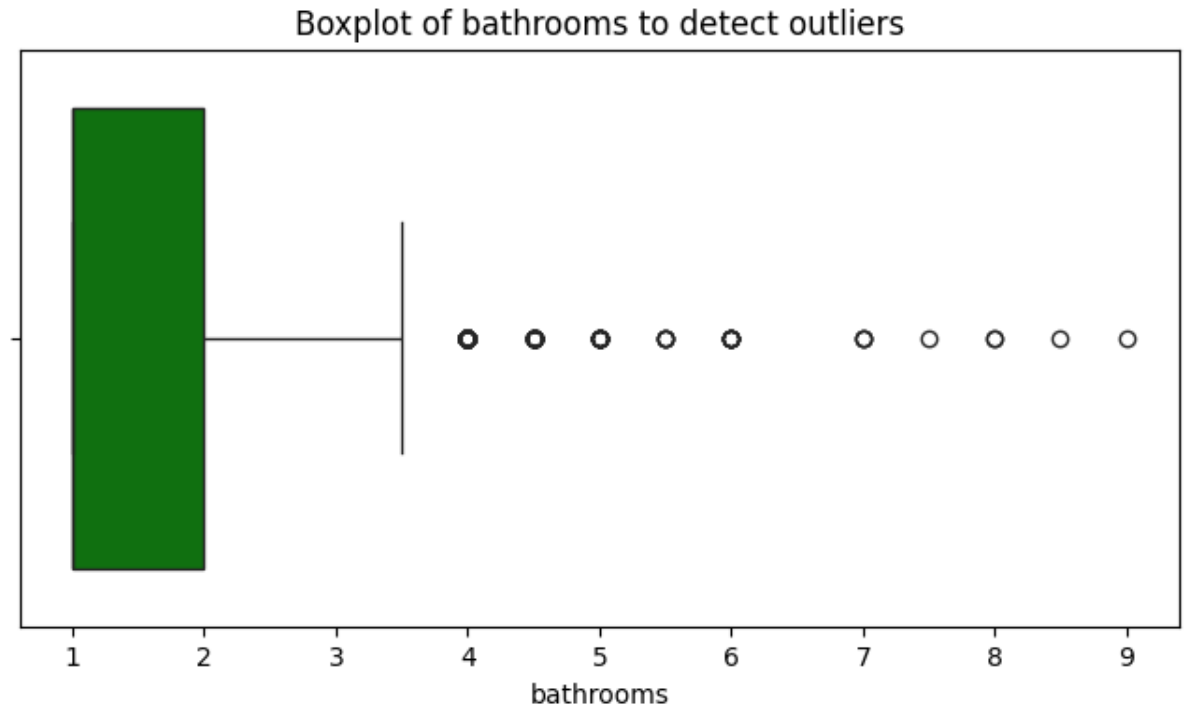

Figure 8: Box-plot of bathrooms

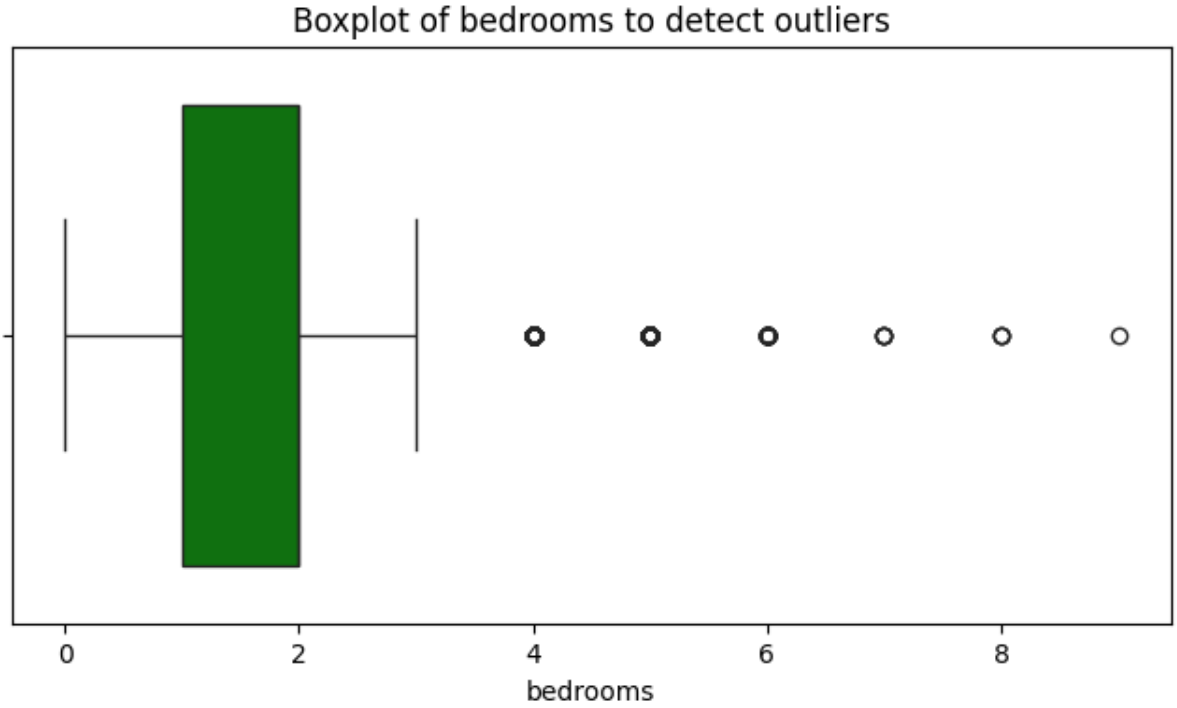

Figure 9: Box-plot of bedrooms

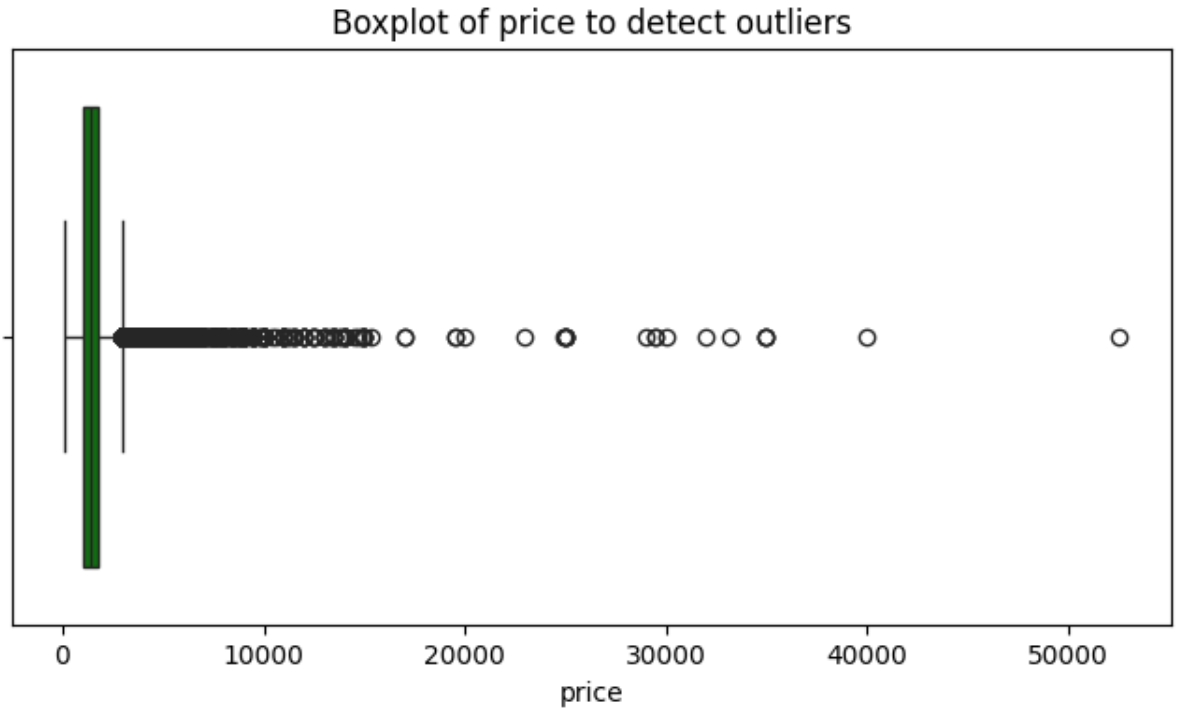

Figure 10: Box-plot of price

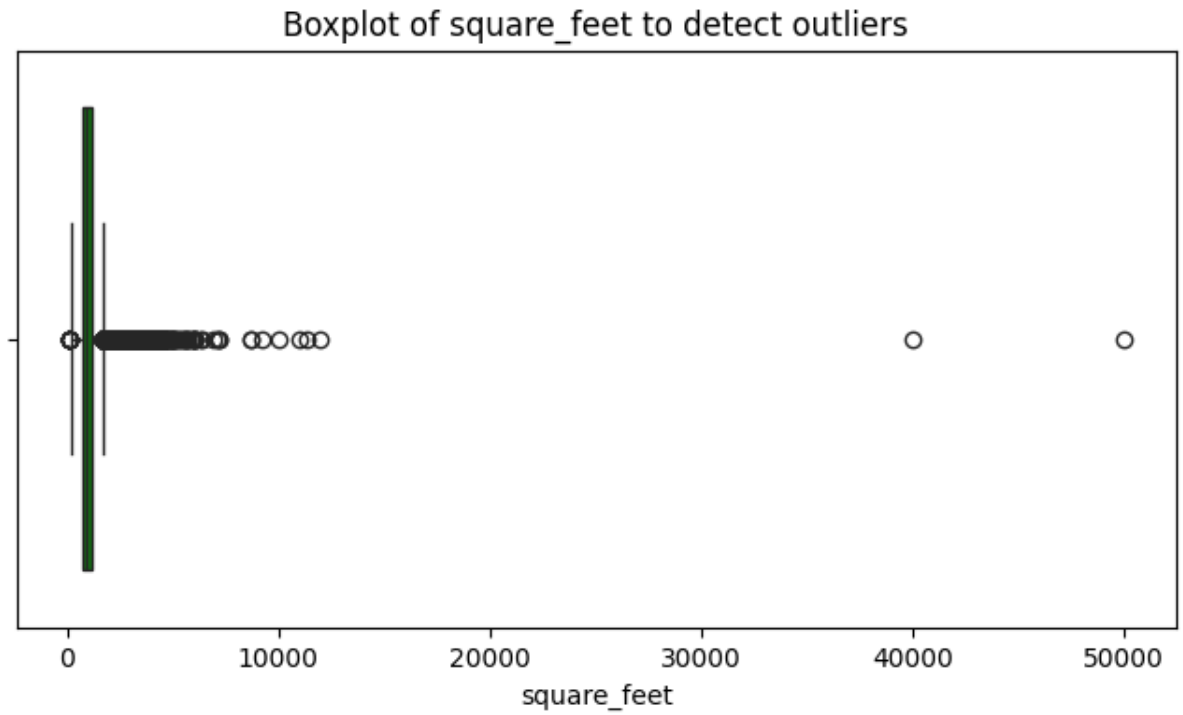

Figure 11: Box-plot of square feet

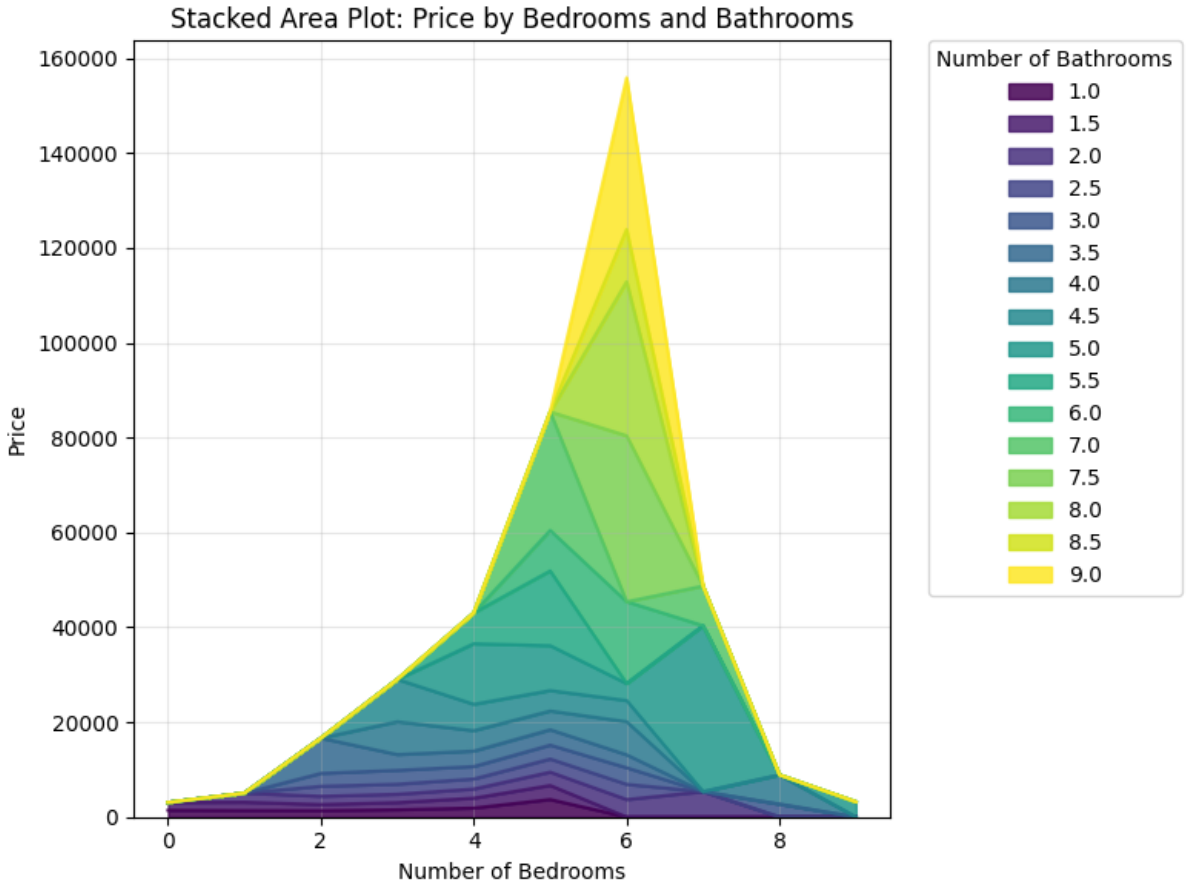

Figure 12: Price by bedrooms and bathrooms

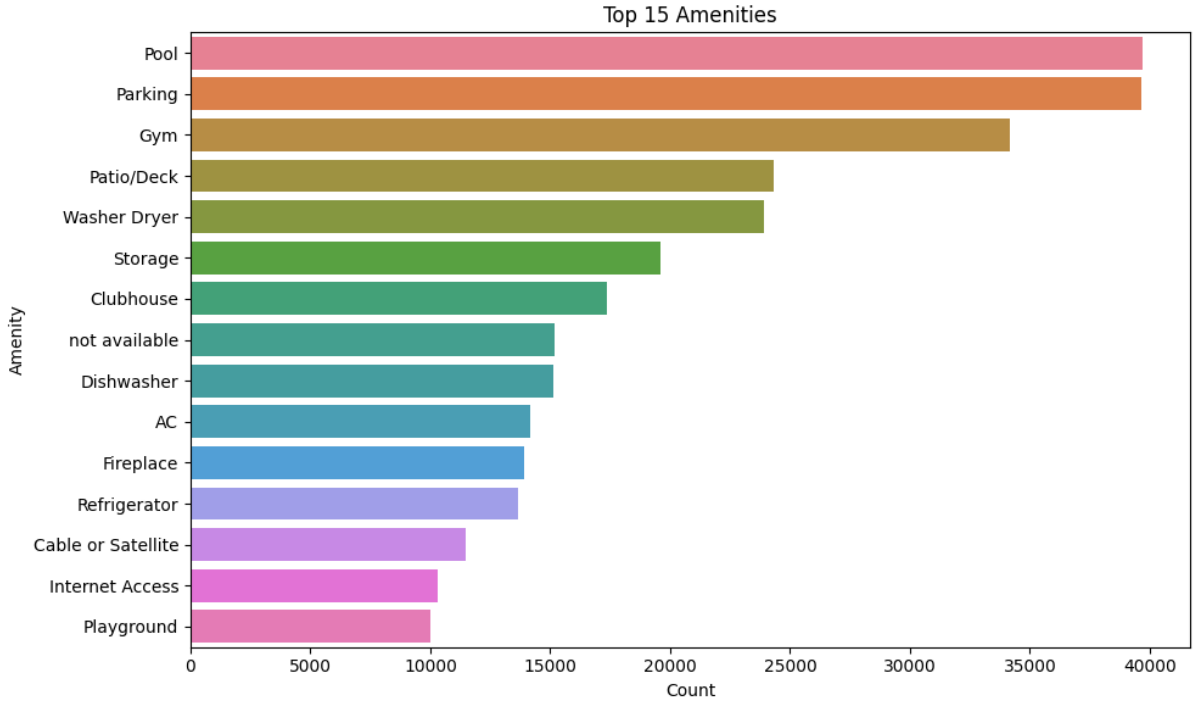

Figure 13: Top 15 Amenities by count

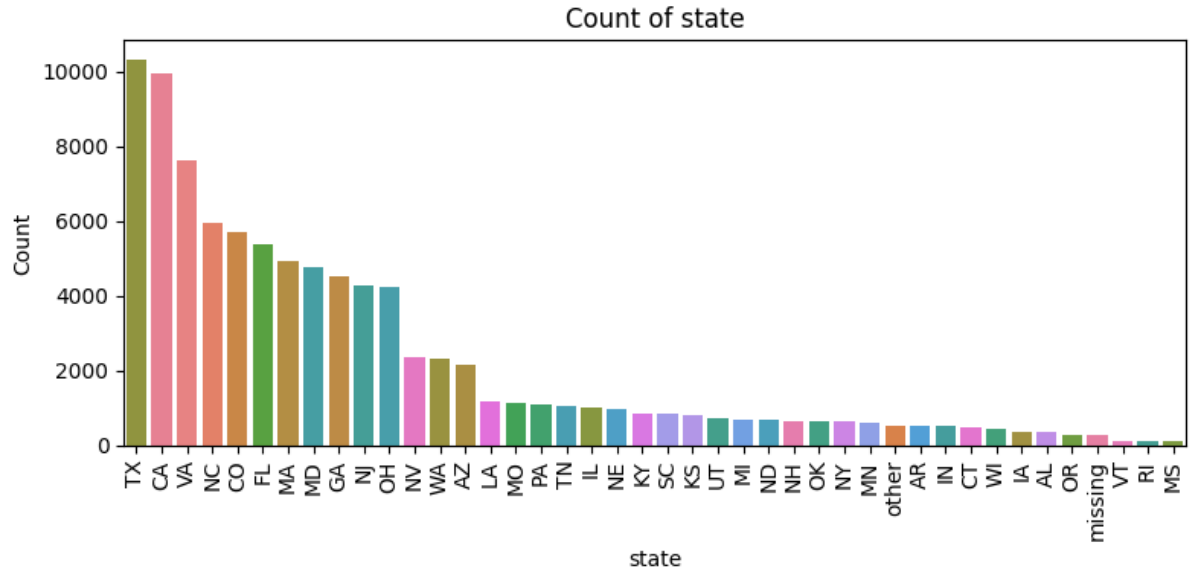

Figure 14: Count of listings by state

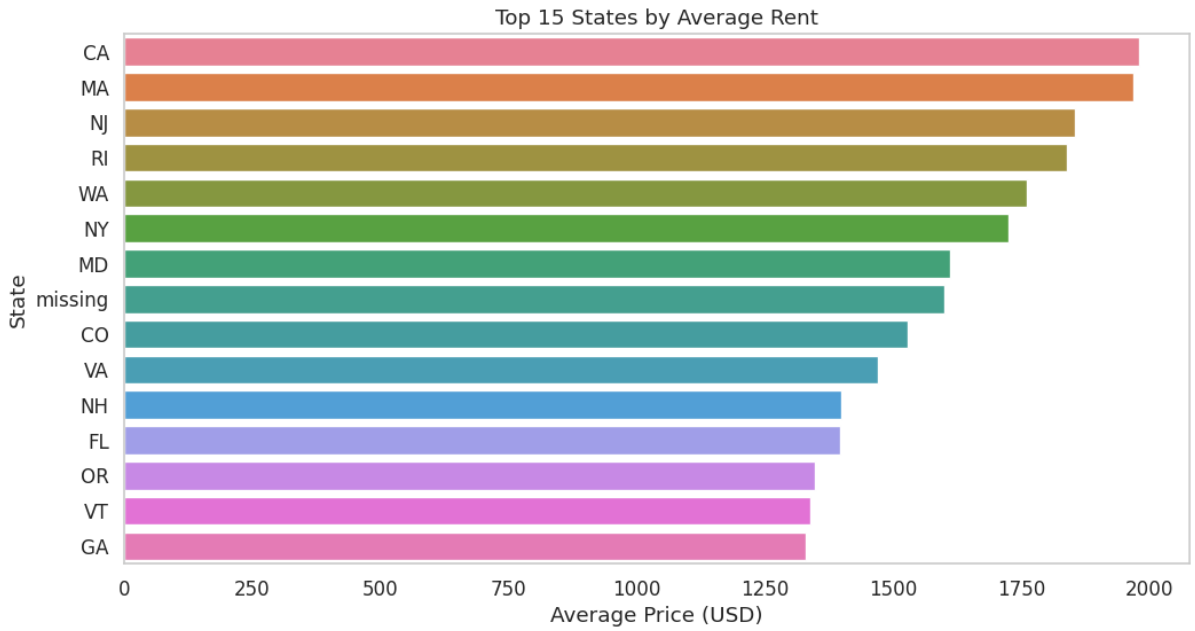

Figure 15: Top 15 States by Price

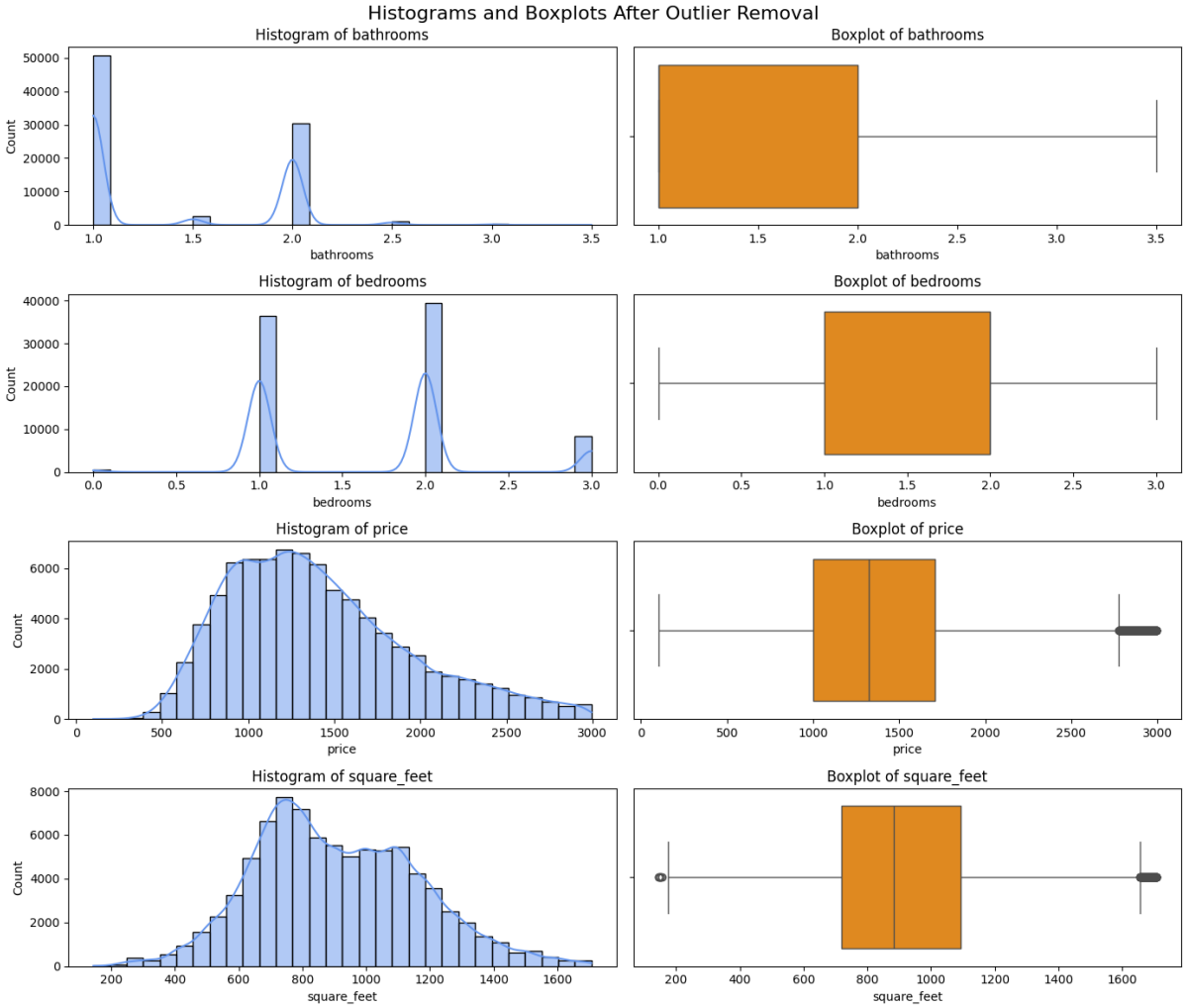

Figure 16: Box-plot and Histograms after Outlier Removal

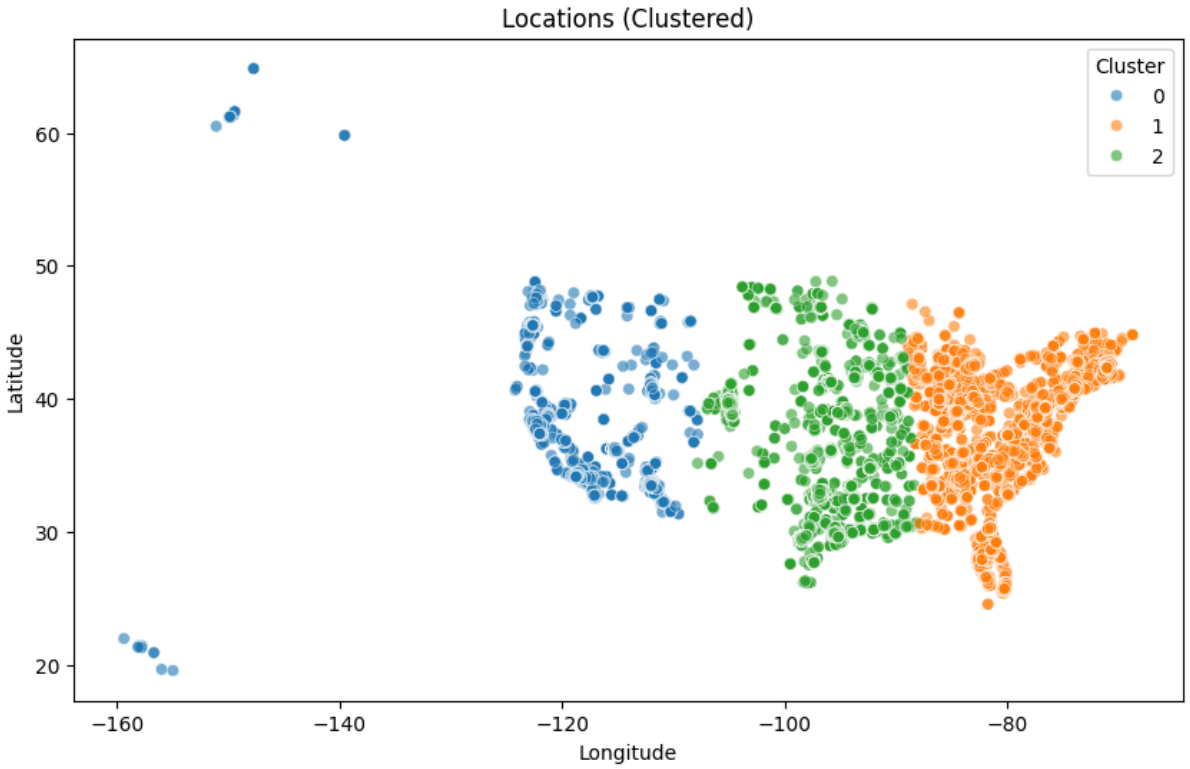

Figure 17: K-Means Clusters based on location

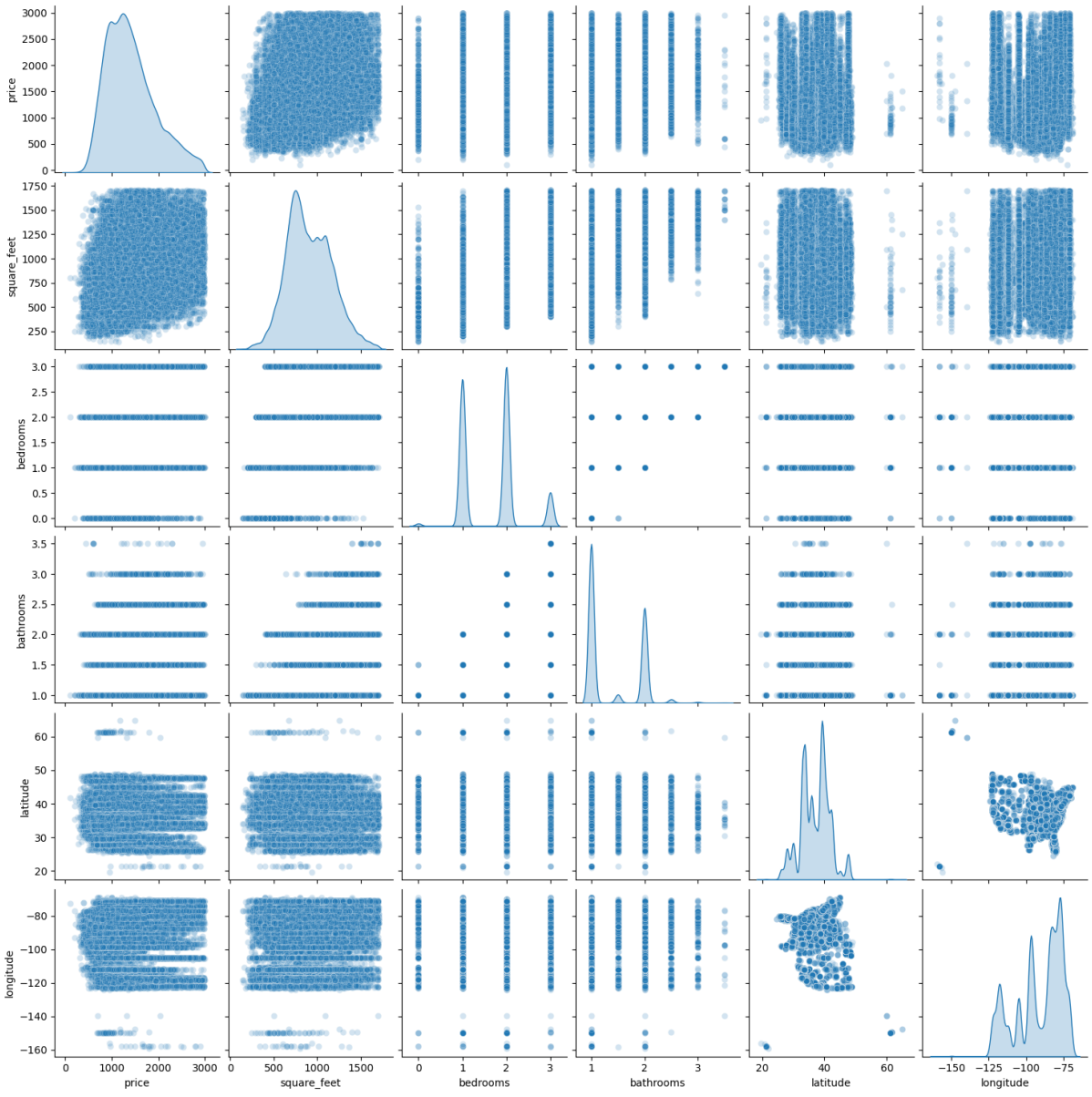

Figure 18: Pairplot of listings

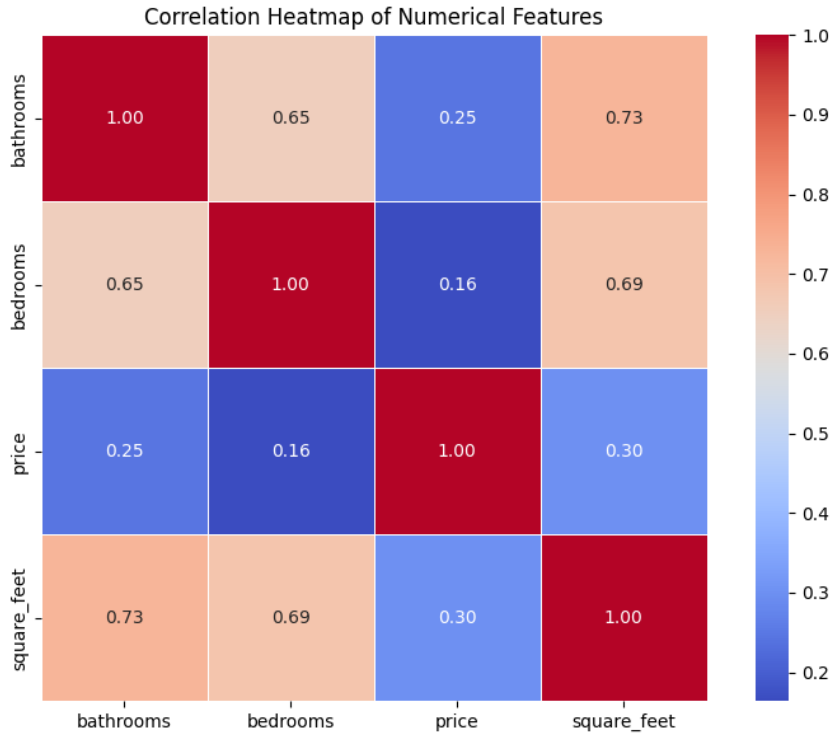

Figure 19: Correlation Heatmap

B Dimensionality Reduction

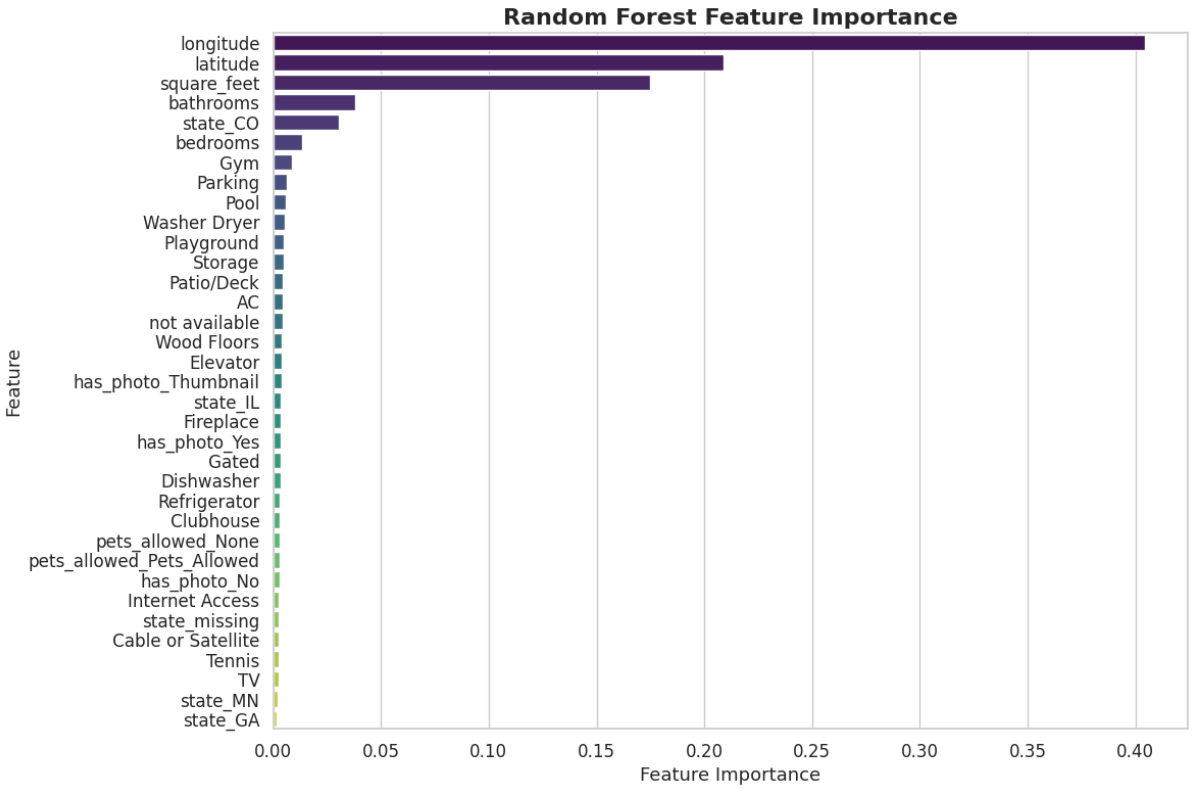

Figure 20: Random Forest Feature Importance

C Regression Analysis

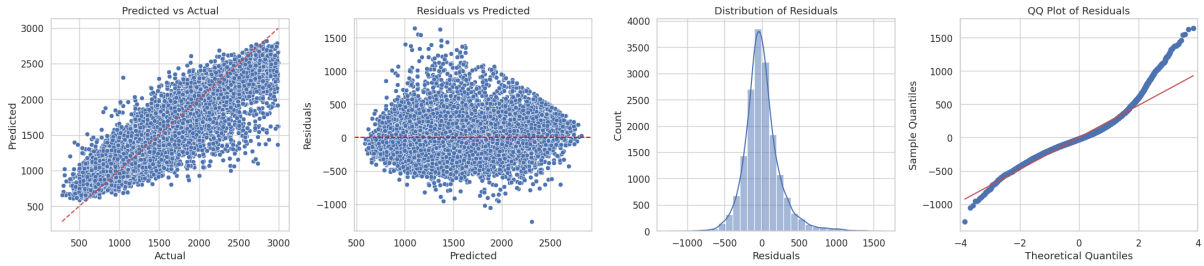

Figure 21: Random Forest Regression Analysis

D Classification Analysis

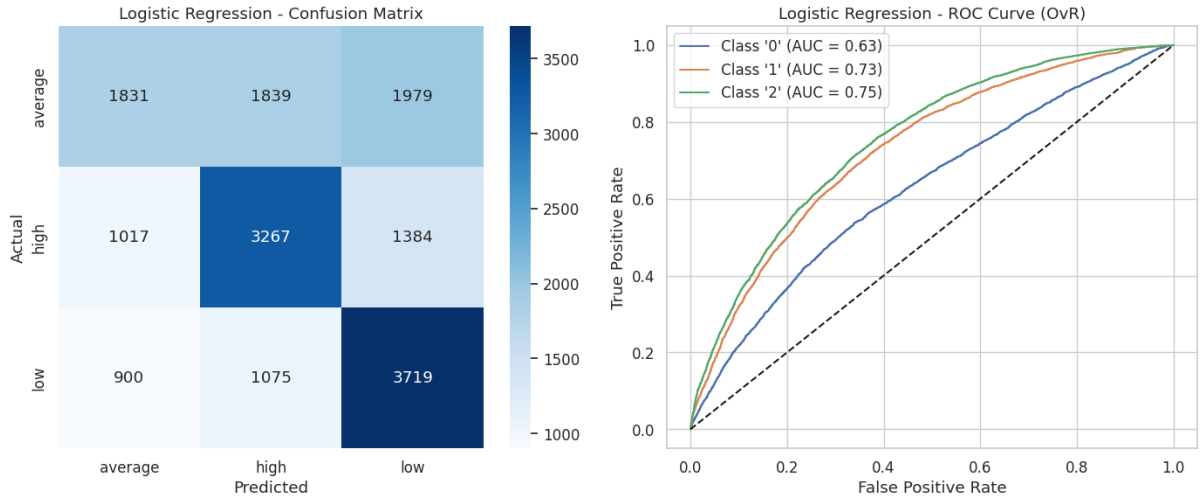

Figure 22: Logistic Regression - Base Model

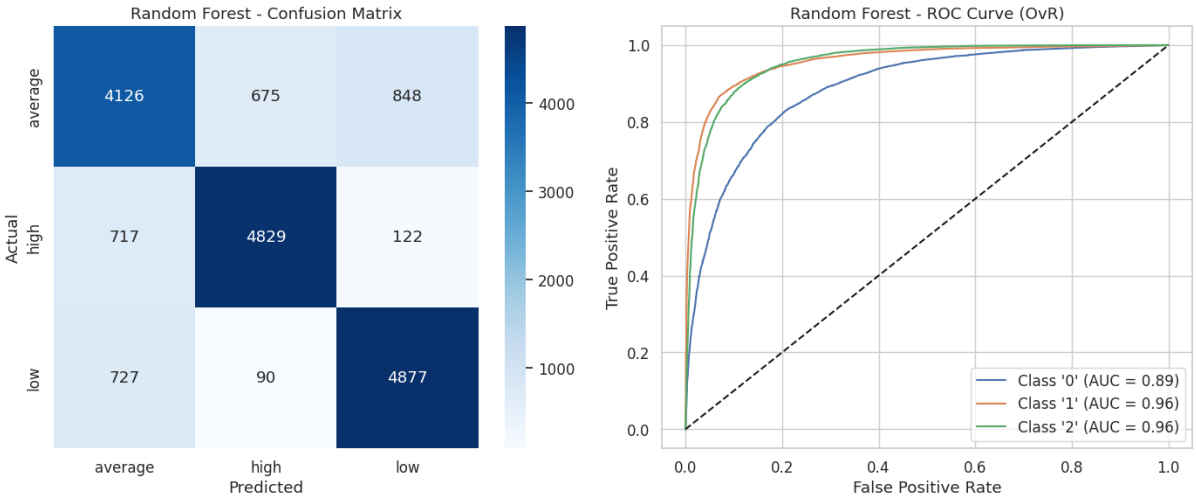

Figure 23: Random Forest - Base Model

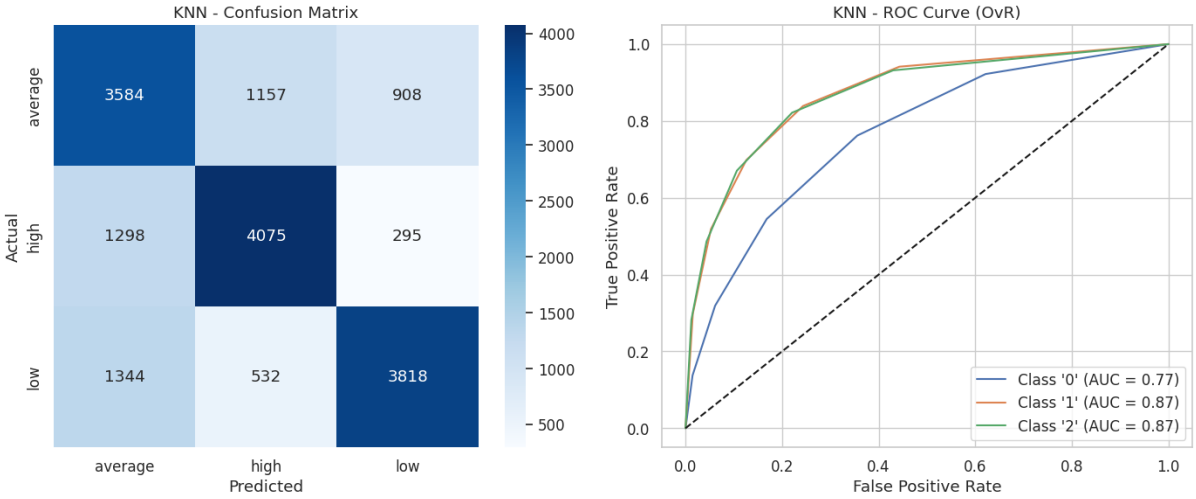

Figure 24: KNN - Base Model

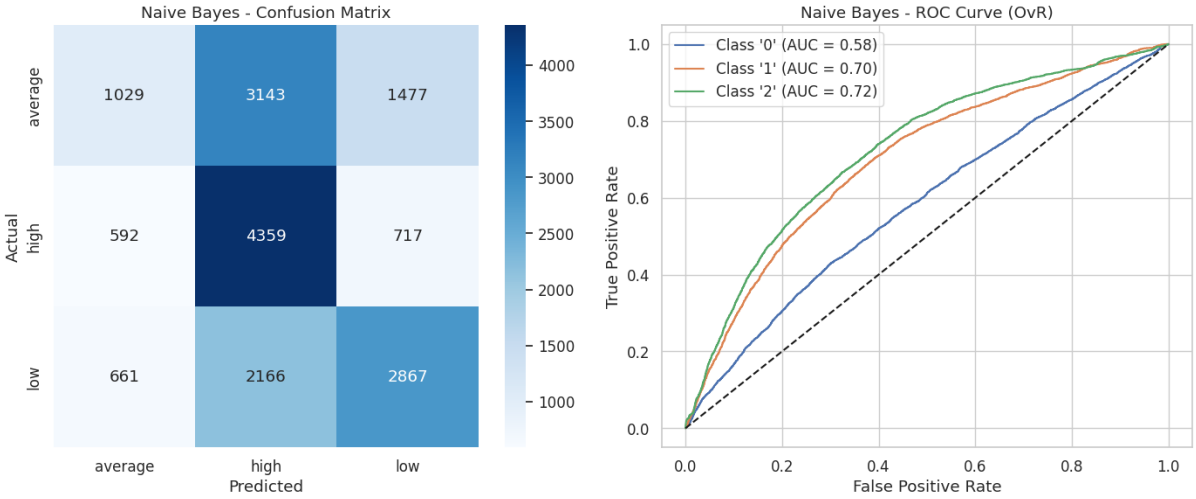

Figure 25: Naive Bayes - Base Model

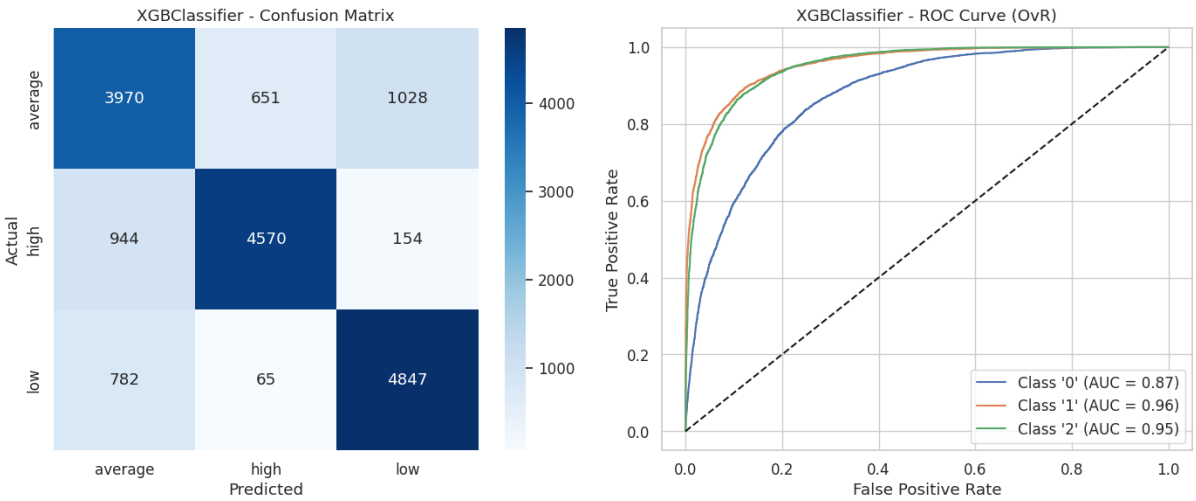

Figure 26: XGBClassifier - Base Model

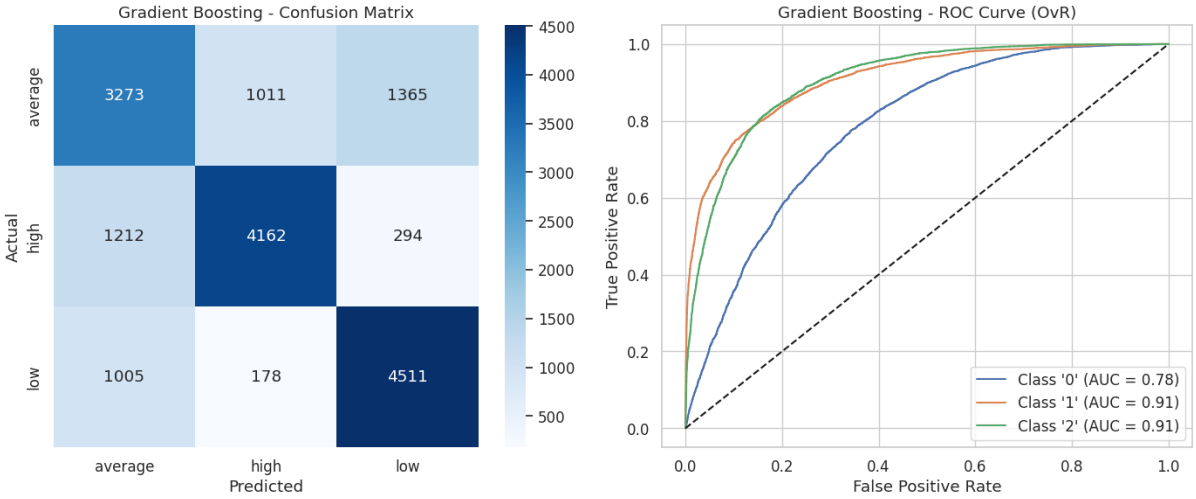

Figure 27: Gradient Boosting - Base Model

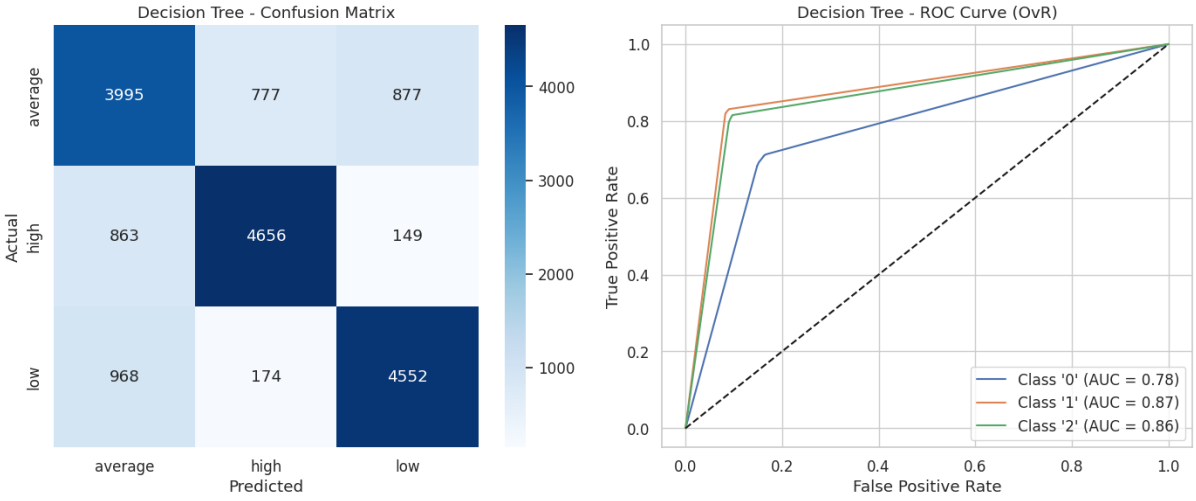

Figure 28: Decision Tree - Base Model

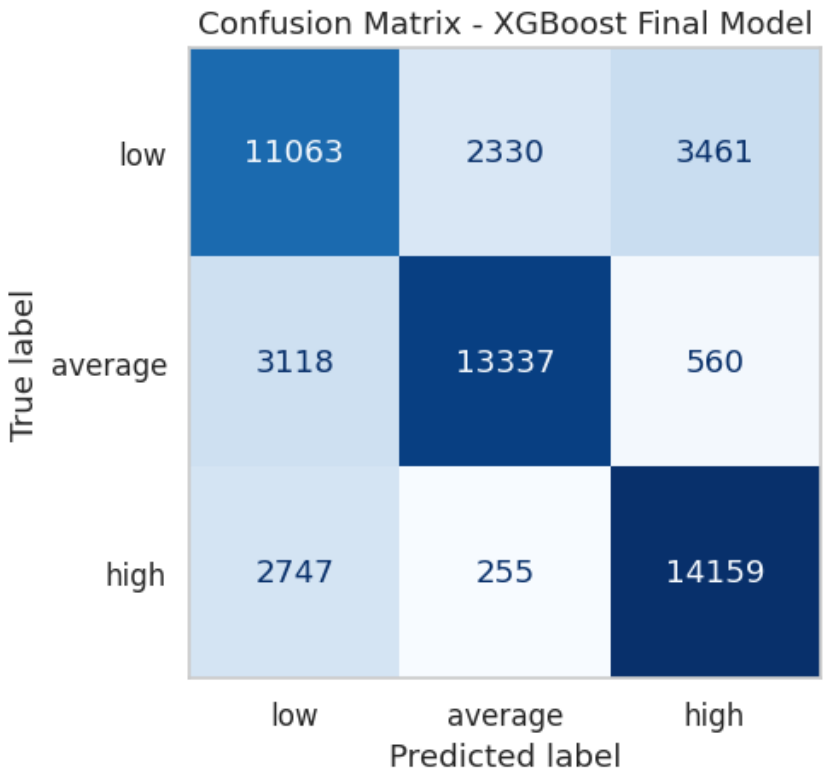

Figure 29: Error analysis of the final XGBoost model.