Predicting Housing Prices using Machine Learning Models

Caleb Miller, Hashim Afzal
College of Computing & Informatics, Drexel University, Philadelphia PA 19104, USA
March 10, 2025
GitHub Repository

Abstract— In this study, we leveraged machine learning techniques with PySpark to develop a regression model for predicting housing prices in King County, WA. We trained multiple models, including Linear Regression, XGBoost Regression, Decision Tree, and MLP Regression, and evaluated their accuracy in predicting housing prices. Among these, XGBoost performed the best, achieving an R^2 of 0.8964 and an RMSE of $135,699.62. Through exploratory data analysis, data preprocessing, and feature engineering, we identified key factors influencing property values and addressed challenges such as outliers, skewed distributions, and multicollinearity. This research demonstrates the effectiveness of PySpark in efficiently handling large datasets, as well as the power of machine learning in housing price prediction. Our findings suggest that incorporating a more geographically diverse dataset with broader features could enhance model robustness and accuracy, making predictions more applicable to a wider range of real estate markets and buyers.

Keywords—PySpark, Machine Learning, Real Estate, Regression

1 INTRODUCTION

Accurate real estate valuation is crucial for investors seeking profitable returns and families looking for fair property prices. Traditionally, home valuation relied on manual appraisals and comparative market analyses, which can be subjective and time-consuming. However, with the advent of machine learning, predictive models can analyze large datasets to identify pricing patterns and improve valuation accuracy [2].

This study focuses on developing a regression model with PySpark to predict housing prices in King County, Washington. By leveraging historical housing data, we identify key factors influencing property values and provide insights for homebuyers, real estate professionals, and investors. Our findings demonstrate the potential of machine learning in real estate pricing and establish a framework that can be adapted to broader housing markets.

2 DATASET

For this project, we downloaded the "House Sales in King County" dataset from Kaggle [1]. The dataset contains 21,614 house sale records from Seattle and surrounding areas [3]. This provides a substantial amount of data points, which will strengthen our model's predictive power by capturing diverse housing trends. The data covers multiple types of variables, such as:

- Transaction Details: transaction id, date of sale, price
- Property Characteristics: number of bedrooms, number of bathrooms, square footage (living area), square footage (entire lot), number of floors, whether or not the property was waterfront, a view rating, a condition rating, a construction grade, square footage (above ground), square footage (basement), when the house was built, if and when a house was renovated
- Neighborhood Characteristics: square footage of the 15 closest neighbors

(living area), square footage of the 15 closest neighbors (entire lot)
- Geospatial Data: zip code, latitude, longitude

With its vast array of property-related features, this dataset provides a strong foundation for predictive modeling by capturing key factors that influence housing prices.

If you would like to access this dataset, it can be found at the following link:

https://www.kaggle.com/datasets/harlfoxem/housesalesprediction [1]

3 EXPLORATORY DATA ANALYSIS

3.1 Distribution Analysis

After collecting the data, Exploratory Data Analysis was performed using PySpark to examine the distribution of our features, detect outliers, and ensure data usability for modeling. Initial analysis demonstrated several variables exhibiting a skewed distribution, with variables like price, bedrooms, square footage (living area), square footage (above ground), and square footage (entire lot) all being skewed to the right.

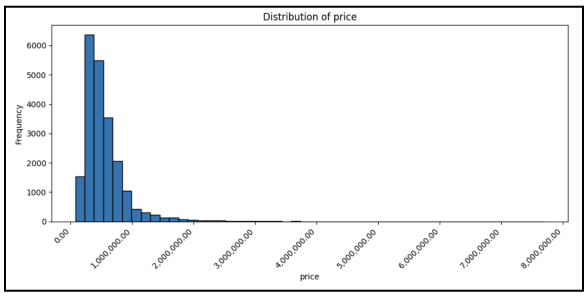

Figure 1: Distribution of Price

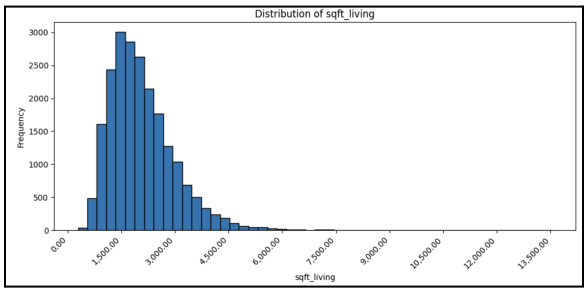

Figure 2: Distribution of Square Footage (Living Area)

The skewed nature of these variables must be addressed before modeling to prevent high-value homes from distorting predictions.

3.2 Outlier Detection

Outliers were identified using the interquartile range (IQR), flagging values exceeding 1.5 times the IQR. One notable anomaly was a property listed with 33 bedrooms, likely due to a data entry error. Even if accurate, such a home is not representative of typical properties.

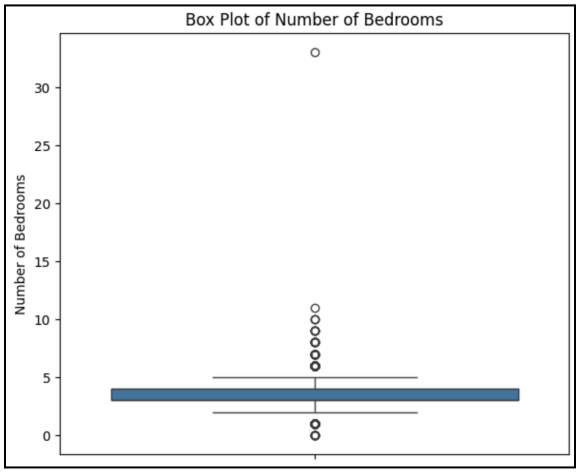

Figure 3: Box Plot of Bedrooms

When examining price, we also found several homes sold for over $5 million. These homes could also skew modeling and would have to be addressed.

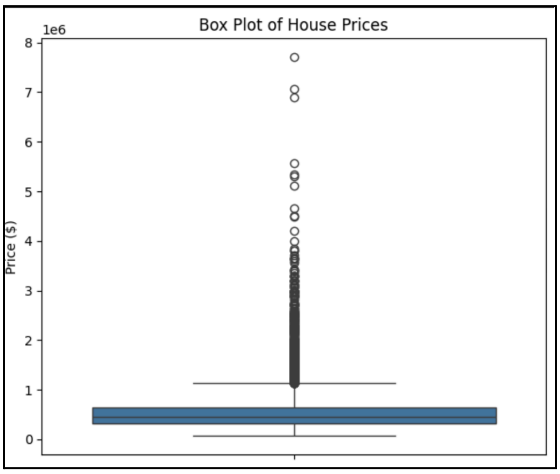

Figure 4: Box Plot of Price

Variables like square footage (entire lot), square footage of the 15 closest neighbors, view rating, and construction grade all also had a significant amount of outliers, and would need to be addressed before moving on to modeling.

Feature	Outlier Count	Outlier Percentage
Square Footage (Entire Lot)	2425	11.2%
Square Footage of the 15 Closest Neighbors	2194	10.1%
View rating	2124	9.8%
Construction Grade	1911	8.8%
Price	1146	5.3%

Table 1: Outlier Counts and Percentages

If left unaddressed, these extreme values could introduce bias, distort model predictions, and lead to overfitting, where the model captures noise rather than meaningful patterns. This could result in inaccurate valuations, especially for mid-range properties. To ensure reliable predictions, we later apply preprocessing techniques to mitigate the influence of these outliers.

3.3 Correlation and Multicollinearity Analysis

A correlation matrix was then constructed to assess relationships between variables. Upon examination, we found a high correlation in size-related features, especially between square footage (living area) and square footage (above ground) (0.86), which is expected since homes without basements will have identical values in both columns. Similarly, square footage (living area) is highly correlated with price (0.67), reinforcing the idea that larger homes tend to be more expensive. Grade (0.70) and bathrooms (0.55) also show strong positive relationships with price, suggesting that higher-quality construction and additional bathrooms significantly impact home values. House age (-0.08) negatively correlates with price, and years since renovation (-0.20) shows that more recent renovations are associated with higher values. These correlations align with expectations, confirming key pricing factors while also highlighting how property updates influence home valuations.

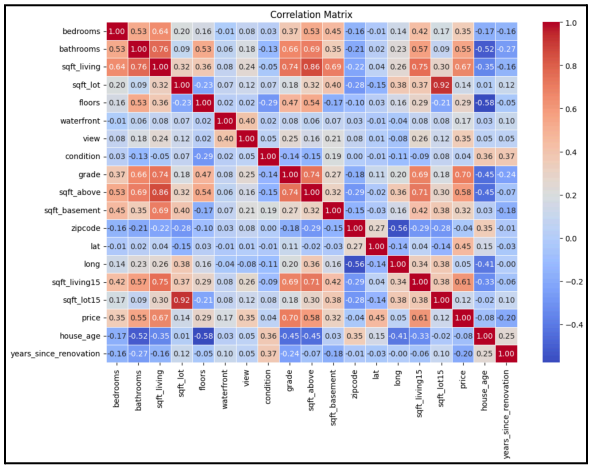

Figure 5: Initial Correlation Matrix

To further investigate multicollinearity, we looked at all numerical variables, paying close attention to the strong correlation between square footage (living area) and square footage (above ground) (0.86). Given their similarity, it was important to determine whether both provided unique value or if one could be removed to reduce redundancy. To quantify multicollinearity across the dataset, we calculated Variance Inflation Factor (VIF) scores, identifying features with excessively high VIF values that could negatively impact model stability and interpretability. Features with high VIF scores were considered for removal to ensure a more reliable predictive model.

	Variable	VIF
1	bedrooms	1.493879
2	bathrooms	2.376580
3	sqft_living	44.383175
4	sqft_lot	5.779807
5	floors	1.895949
6	waterfront	1.467318
7	view	1.904452
8	condition	1.304073
9	grade	2.827467
10	sqft_above	35.726492
11	sqft_basement	7.837224
12	zipcode	2.001825
13	lat	1.382941
14	long	2.429246
15	sqft_living15	2.395798
16	sqft_lot15	5.809883
17	house_age	1.391459
18	years_since_renovation	1.559302

Figure 6: Initial VIF Scores

As expected, square footage (living area) and square footage (above ground) were found to be very correlated with one another. Generally, VIF values above 10 should be investigated and in our case we opted to drop square footage (above ground) due to redundancy. The final VIF calculations stayed within the acceptable range of 5.

	Variable	VIF
1	bedrooms	1.493149
2	bathrooms	2.370858
3	sqft_living	4.257555
4	sqft_lot	5.764796
5	floors	1.802998
6	waterfront	1.463827
7	view	1.897362
8	condition	1.301174
9	grade	2.644789
10	sqft_basement	1.795035
11	zipcode	2.000287
12	lat	1.382078
13	long	2.420938
14	sqft_living15	2.382789
15	sqft_lot15	5.808214
16	house_age	1.380438
17	years_since_renovation	1.544849

Figure 7: VIF Scores Post-Adjustments

After finalizing our correlation matrix by removing highly collinear variables, we found that the overall relationships between features remained unchanged. Square footage (living area) (0.67), grade (0.70), and bathrooms (0.55) continued to show strong positive correlations with price, while house age (-0.08) and years since renovation (-0.20) retained their weaker negative relationships. This confirms that eliminating multicollinearity did not alter key pricing drivers, ensuring that our dataset remains both statistically sound and interpretable for predictive modeling.

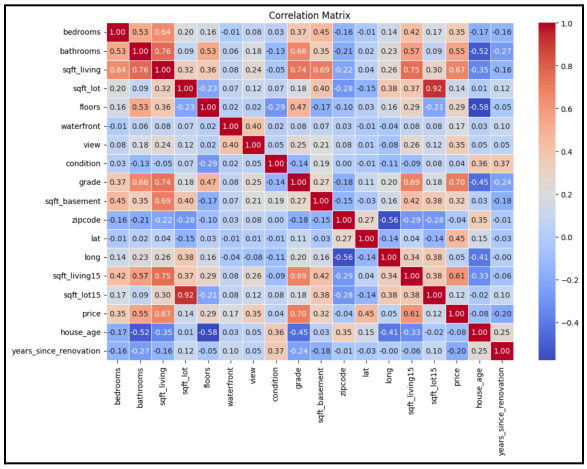

Figure 8: Final Correlation Matrix

4 METHODOLOGY

4.1 Data Preprocessing

Before modeling, we first cleaned the data to ensure its integrity. Using PySpark, we verified that there were no missing values, except for square footage (basement), where null values appeared for houses without a basement. To address this, we replaced the null values with 0, accurately reflecting the absence of a basement. Then we converted the variables to their appropriate data types and began dropping unnecessary features. These included:

- Transaction ID - This variable is just a unique identifier. This does not influence price.
- Latitude - Dropped due to redundancy with zip codes.
- Longitude - Also dropped due to redundancy with zip codes.

These steps ensured that our models were built on clean, accurate, and non-redundant data.

4.2 Feature Engineering

To help our models better capture trends in the data, we performed feature engineering by creating new features from existing ones. First, we converted the year built into the house's age, changing the range from 1900–2015 to 10–125. This transformation reduced the scale of the values, making it easier for the model to distinguish differences between data points. Similarly, we calculated the years since renovation instead of using the renovation year directly. However, some values for renovation year were 0, indicating the house had never been renovated. To handle this, we replaced 0s with nulls before performing the calculation. Finally, we filled these null values with the house's age, effectively treating unrenovated homes as having been in their original condition since construction.

We also needed to address our issues with outliers and skewed data. To address the outliers, we applied winsorization to the following variables:

- Bedrooms, bathrooms, square footage (living area), square footage (entire lot), square footage (basement), square footage 15 (living area), square footage 15 (entire lot)

We replaced values in these columns that fell at the 100th percentile with the 99th percentile value to mitigate the impact of extreme outliers. For example, one house in the dataset had 33 bedrooms, which was at the 100th percentile. We replaced it with the 99th percentile value of 6, demonstrating how this technique effectively removes anomalies while preserving the overall data distribution.

To correct the skewed data, logging was applied to the following variables:

- Price, square footage (living area), square footage (entire lot), square footage (basement), square footage 15 (living area), square footage 15 (entire lot)

This transformation helped normalize the distribution of these variables, reducing the impact of extreme values and making patterns in the data more distinguishable for the model. By compressing large values while preserving relative differences, the log transformation improved model performance.

4.3 Modeling and Results

Model	R^2 Score	RMSE ($)
XGBoost	0.8964	135,699.62
MLP Regression	0.8247	191,888.17
Linear Regression	0.8148	193,438.71
Decision Tree	0.7732	140,705.00

Table 2: Model Performance

4.3.1 Decision Tree

First, we will discuss our decision tree regression model. This model was utilized as a non-linear alternative to linear regression. Decision Trees are seen as being quite effective in identifying relationships between variables and are seen as more complex than standard linear regression. The model was trained to a max depth of 10, and bins were set to 32, balancing computation required and complexity. In the end however, the model did not perform as well as expected.

Model Performance:

R^2 Score: 0.7732

Root Mean Squared Error (RMSE): $140,705.00

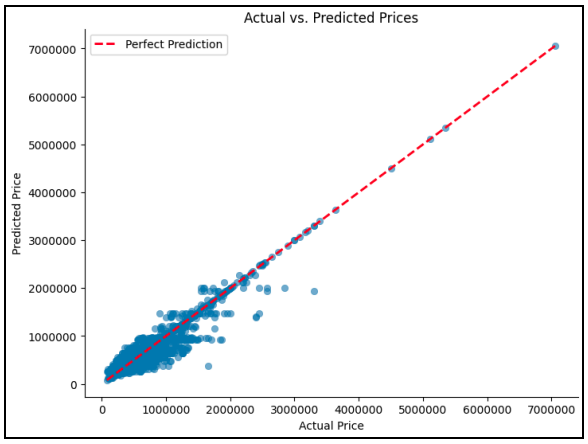

Figure 9: Decision Tree Actual vs Predicted

While the model seemed to fit the training data fairly well, it failed to explain much of the overall variance in housing prices. This was highlighted by the model holding the second lowest RMSE value, but when looking at the second performance metric, R^2, it is clear the model was overfitting. Its inability to generalize to new data makes this model unsuitable for pricing prediction.

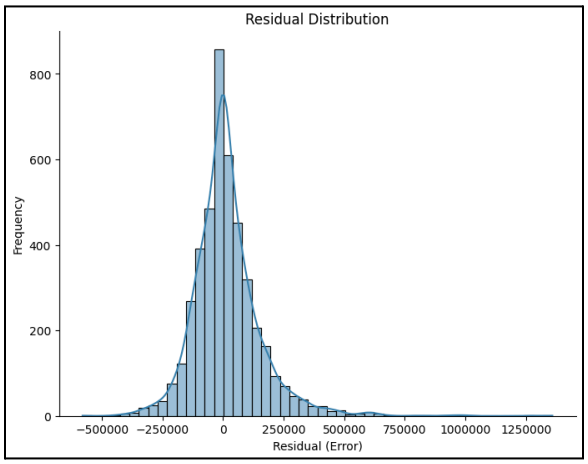

Figure 10: Decision Tree Residual Distribution

4.3.2 Linear Regression

As the least complex model in our project, Linear Regression was trained to serve as a baseline for comparison. Despite its simplicity, it performed third best out of four models, outperforming the decision tree, which was more prone to overfitting. This highlights that when data follows a mostly linear trend, simpler models can generalize better than more complex ones.

Model Performance:

R^2 Score: 0.8148

Root Mean Squared Error (RMSE): $193,438.71

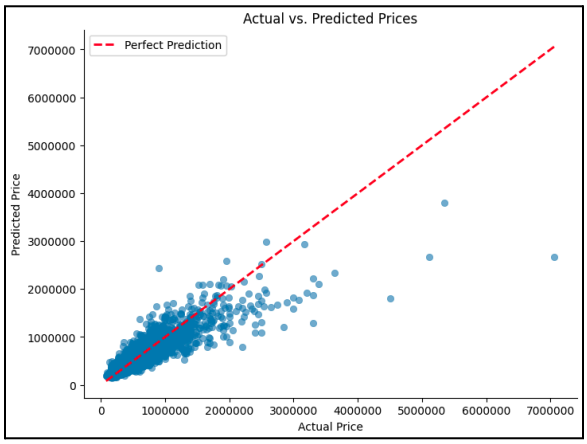

Figure 11: Linear Regression Actual vs Predicted

The model captured overall housing trends well, making reasonable predictions for mid-range properties. However, it struggled with extreme values, particularly high-end homes, where it tended to underpredict prices. This limitation suggests that price variations in luxury homes involve more complex, nonlinear relationships that linear regression cannot fully capture.

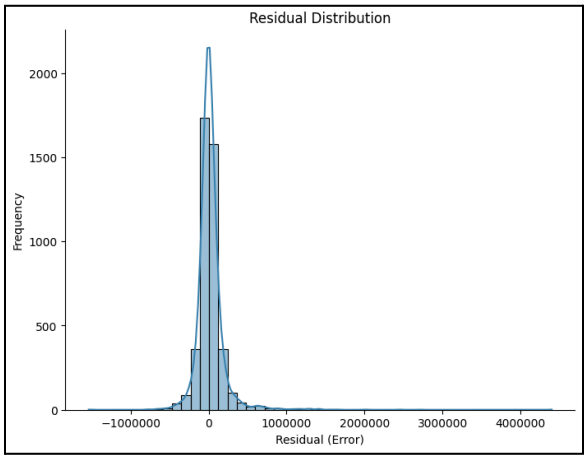

Figure 12: Linear Regression Residual Distribution

Despite these challenges, the model provided a solid benchmark for evaluating more advanced approaches. Its ability to outperform the decision tree suggests that increased model complexity does not always lead to better results. Moving

forward, we explored ensemble methods, such as XGBoost, and nonlinear models, like MLP Regression, to assess whether they could improve predictive accuracy while maintaining generalizability.

4.3.3 MLP Regressor

To capture complex patterns in the data, we trained a Multi-Layer Perceptron (MLP) Regression model, leveraging a neural network with two hidden layers (64 and 32 neurons, respectively) and ReLU activations. The model was optimized using the Adam optimizer and trained with the Huber loss function, which helped mitigate the impact of outliers. PySpark's functionality does not include an MLP Regressor, so we had to convert to a pandas DataFrame and work with TensorFlow for training, then convert back to PySpark for analysis.

Model Performance:

R^2 Score: 0.8247

Root Mean Squared Error (RMSE): $191,888.17

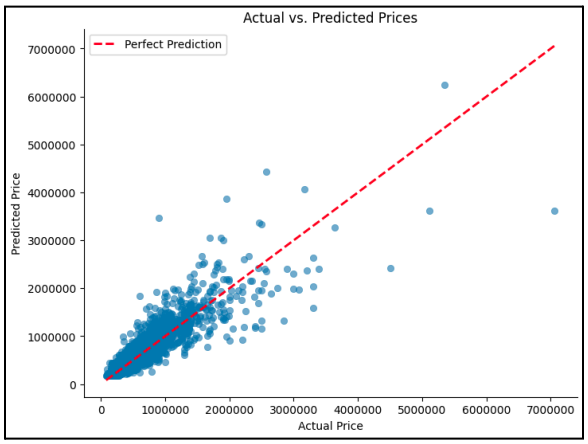

Figure 13: MLP Regression Actual vs Predicted

Among the four models we tested, MLP Regression achieved the second-best performance, outperforming both Linear Regression and the Decision Tree but falling short of XGBoost. Its ability to learn nonlinear relationships contributed to improved predictive accuracy over simpler models.

While MLP captured complex interactions well, it still struggled with extreme price values, particularly high-end properties. Compared to XGBoost, which achieved the best performance, MLP required significantly more computational resources and hyperparameter tuning while still being outperformed.

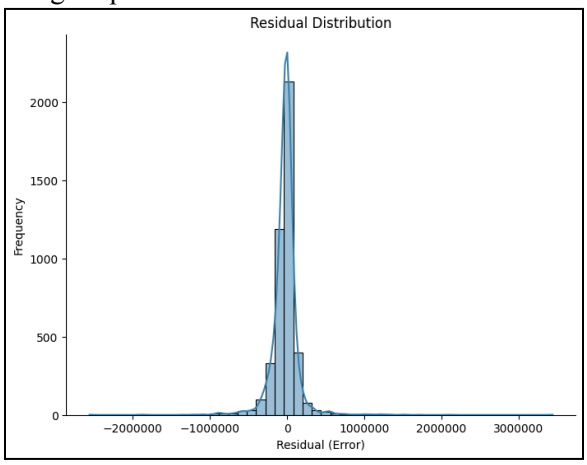

Figure 14: MLP Regression Residual Distribution

Although MLP was the second-best model, this comparison highlights that for structured tabular data like housing prices, ensemble methods like XGBoost might offer a better balance of accuracy and efficiency than deep learning models.

4.3.4 XGBoost

XGBoost, or Extreme Gradient Boosting, was utilized as a more sophisticated alternative to decision tree regression. XGBoost leverages boosting, where multiple decision trees or weak learners are combined iteratively in order to minimize errors. In order to maximize performance, we implemented grid search to optimize hyperparameters and utilized a learning rate of 0.3, max depth of 3 and 150 estimators.

Model Performance:

R^2: 0.8964

Root Mean Squared Error (RMSE): $135,699.62

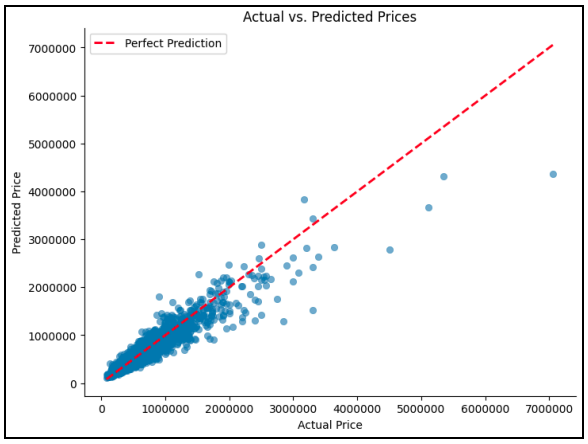

Figure 15: XGBoost Actual vs Predicted

Among all models, XGBoost had the best performance across both metrics, and was able to capture non linear relationships better than basic models such as linear regression and decision trees, but was also better at generalizing and capturing outliers than a more complex model like MLP Regression. Unlike a standard decision tree, XGBoost incorporates regularization, preventing it from overfitting.

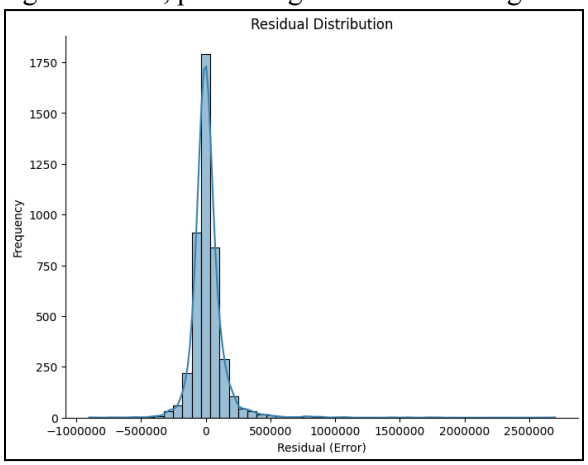

Figure 16: XGBoost Residual Distribution

While XGBoost does require more computational power than linear regression and decision trees, its accuracy in both R^2 and RMSE justify its utilization in tasks such as housing price prediction while still being less computationally expensive than a deep learning model like our MLP Regression. With the correct data, XGBoost was seen as very effective in this sort of task.

5 CONCLUSION AND FUTURE WORKS

This project highlighted the effectiveness of machine learning models in predicting housing prices. Despite its simplicity, Linear Regression performed fairly well and is still a great tool to establish a baseline for more complex modeling approaches. While decision trees are appropriate models for certain tasks, we found that they were not suitable for the prediction of housing prices in King County, Washington. Price prediction requires a model to generalize well, and decision tree models are simply too prone to overfitting. The MLP Regressor was the only model not to be coded in PySpark, as PySpark did not support the use of the model. This may highlight a limitation of deep learning models with big data platforms, as there is limited support. Overall, XGBoost performed best in both R^2 and RMSE, and its connectivity with PySpark makes it a suitable model for big data analysis.

One limitation of this study was the accessibility of real estate data for researchers as companies like Zillow and Redfin limit access to their API without a paid access key. With a larger training set, and access to data nationwide as opposed to specific to King County, this modeling framework could serve as a comprehensive approach to predicting housing prices across the United States. More future work includes incorporating additional features such as crime statistics, interest rates and weather information such as average temperature, average precipitation and sun exposure. It is also necessary to include up-to-date data, as this dataset includes only 2014 to 2015 which reduces the relevancy of these predictions. Expansion to include more features, more current data, and incorporating a dataset to include properties all over the U.S. could make PySpark's distributed computing capabilities effective as researchers could overcome the computational limitations traditional data processing methods possess. By expanding this research, machine learning models could be adapted for broader real estate markets, enabling investors and homebuyers to make more data-driven decisions.

REFERENCES

[1] Harlfoxem. (2015, May). *House sales in king county.* Retrieved 2025, February 23 from https://www.kaggle.com/datasets/harlfoxem/housesalesprediction

[2] Hernes, M., Tutak, P., Nadolny, M. & Mazurek, A. (2024, November 28). *Real estate valuation using machine learning.* ScienceDirect. https://www.sciencedirect.com/science/article/pii/S1877050924023482

[3] Wikipedia contributors. (2025, March 1). *King County, Washington*. Wikipedia. https://en.wikipedia.org/wiki/King_County,_Washington