A Comparative Analysis of Machine Learning Algorithms in House Price Classification

Christian Jayson Juerba[1], Jucris Galleto[2], James Philip Carismal[3]

[123]*University of Mindanao, Matina, Davao City, Philippines*
c.juerba.488016@umindanao.edu.ph, j.galleto.547759@umindanao.edu.ph,
j.carismal.549639@umindanao.edu.ph

***Abstract* – This paper aims to compare the effectiveness of different machine learning algorithms in classi fying house prices into distinct market categories (e.g., budget, affordable, premium and luxury) rather than predicting continuous values. Real estate valuation significantly impacts economic decisions, investment strategies, and housing policies. The paper presents a comparative analysis of various classification algorithms including Neural Networks, Random Forest, XGBoost, Support Vector Machines, K-Nearest Neighbors, and Logistic Regression. The classifiers were evaluated across multiple metrics including accuracy, precision, recall, F1-score, and computational efficiency. The results showed that ensemble methods, particularly XGBoost and Random Forest, outperformed other algorithms across most evaluation metrics. The study contributes to real estate analytics and provides practical insights for property valuation, investment analysis, and automated appraisal systems. The findings can benefit real estate professionals, financial institutions, and policymakers in making informed housing market decisions.**

***Keywords – House Price Classification, Comparative Analysis, Machine Learning Algorithms, Real Estate Analytics, Predictive Modeling, XGBoost, Random Forest, Neural Networks,* Support Vector Machines, K-Nearest Neighbors, Logistic Regression**

1. Introduction

The [1] House price classification plays a significant role in real estate analysis, as it directly affects property valuation, investment decisions, and market assessment. Accurate classification of housing prices enables stakeholders to better understand pricing trends and market behavior. However, traditional statistical and econometric techniques often struggle when applied to large datasets with complex feature interactions and nonlinear relationships among housing attributes such as price, bedroom, bathroom, floor, view, condition, squarer foot living, square foot lot, waterfront, square foot basement, year built and year renovation [2].

[4] With the advancement of machine learning, data-driven approaches have been increasingly applied to real estate problems. Machine learning algorithms are capable of learning patterns from historical housing data and modeling nonlinear dependencies that conventional methods fail to capture. As a result, these techniques have been widely adopted for house price prediction and classification, demonstrating improved accuracy and robustness [5].

[6] Several machine learning models, including ensemble-based methods and neural networks, have shown strong performance in handling structured housing data. Ensemble algorithms such as Random Forest and gradient boosting models are particularly effective due to their ability to reduce overfitting and improve generalization [3]. Neural networks have also been successfully applied in real estate analytics because of their capability to model complex feature relationships [9].

[7] Despite the increasing use of machine learning in housing price analysis, there remains a need for comparative studies that evaluate multiple algorithms under a unified preprocessing and evaluation framework. Such studies provide insights into the strengths and limitations of each algorithm and help identify the most suitable model for house price classification tasks [11].

Objectives

1.1.1 To implement and apply different machine learning algorithms for house price classification.

1.1.2. To preprocess the housing dataset in order to improve classification performance.

1.1.3. To evaluate and compare the performance of the algorithms using standard classification metrics.

2. Methodology

2.1 Data Gathering

The dataset used in this study is *House_Price.csv*, obtained from Kaggle [1]. This dataset contains records of residential properties along with their corresponding sale prices and various property-related attributes. The features describe both structural and physical characteristics of houses, including price, number of bedrooms, number of bathrooms, total living area, lot size, number of floors, and other relevant indicators. The dataset was selected because it is well-structured, widely used in academic research, and suitable for both predictive and classification- based machine learning tasks. It provides sufficient data points to train and evaluate multiple machine learning models for house price classification.

https://www.kaggle.com/datasets/shree1992/housedata/data

2.2 **Data Analysis**

The Exploratory Data Analysis (EDA) was conducted to gain an initial understanding of the dataset and its characteristics. This step involved examining the structure of the data, identifying numerical and categorical features, and checking for potential data quality issues such as missing values and outliers. Descriptive statistics, including minimum, maximum, mean, and standard deviation, were computed to summarize the numerical attributes and understand their distributions.

The target variable, *price*, was further analyzed to observe its distribution and level of skewness. As commonly observed in real estate datasets, house prices exhibited a right-skewed distribution, with a small number of properties having significantly higher values. This observation highlighted the need for appropriate preprocessing techniques to reduce the impact of extreme values and improve model stability.

.

Table 1. House Price Prediction Statistic Structure

	Price	**Bedroom**	**Bathroom**	**Floor**	**Sqrft lot**	**Sqrft Living**
Min	0	0	0	1	6.380000e+02	370
Max	2.659000e+07	9	8	3.5	1.074218e+06	13540
Mean	5.519630e+05	3.4	2.1	1.5	1.485252e+04	2139
Std	5.638347e+05	9	0.78	0.5	3.588444e+04	963

Table 1. Show sample of the Statistic Structure of the Hourse Price Prediction.

To convert the original regression-based problem into a classification task, the continuous house price variable was transformed into categorical classes. This approach allows for easier interpretation of results and aligns with real-world use cases where properties are grouped into price ranges. The house prices were divided into four categories:

- **Budget** – lowest price range
- **Affordable** – lower-middle price range
- **Premium** – upper-middle price range
- **Luxury** – highest price range

The categorization was based on percentile thresholds (25th, 50th, and 75th percentiles) to ensure a relatively balanced class distribution. This method reduces class imbalance and helps improve the performance of classification algorithms used in the study.

Table 2. Distribution of House Price Categories

Category	**Description**	**Price Range**
Affordable	Lower price range	$0.00 - $322,500.00
Budget	Moderate price range	$323,000.00 - $460,886.92
Premium	Higher price range	$461,000.00 - $654,950.00
Luxury	Highest price range	$655,000.00 - $2,005,220.00

Table 2. Show The target variable, house price, was transformed into categorical price classes labeled Affordable, Budget, Premium, and Luxury based on percentile-based thresholds.

2.3 **Data Preprocessing**

Data preprocessing was performed to improve data quality and enhance model performance. Several preprocessing steps were applied, including handling missing values, detecting and treating outliers, and scaling numerical features. Missing values in the dataset were addressed using mean imputation for numerical features. In this approach, missing entries were replaced with the mean value of the corresponding feature. This method ensures that no data records are removed while maintaining the overall statistical properties of the dataset. Mean imputation was chosen due to its simplicity and effectiveness for numerical data [2].

2.3.4 **Handling Outliers**

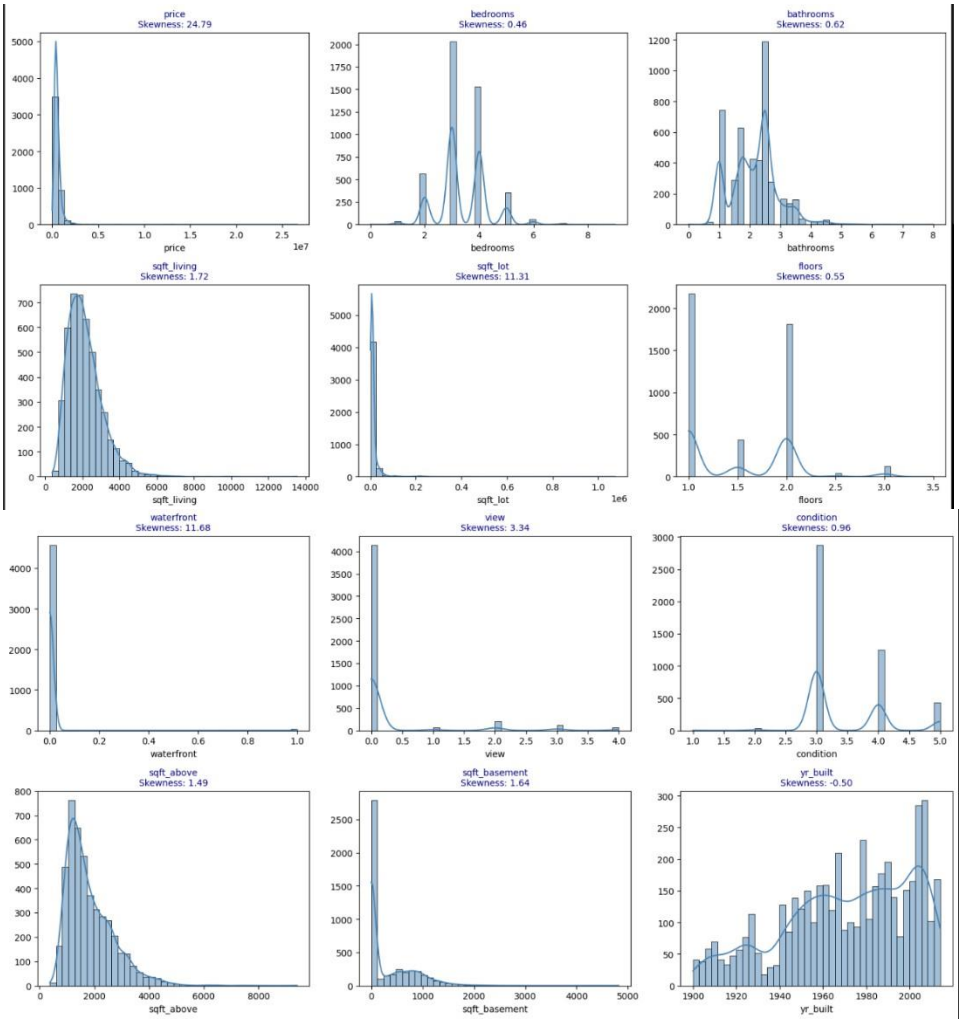
Outliers were identified using the Interquartile Range (IQR) method [3]. For each numerical feature, values below Q1 − 1.5×IQR or above Q3 + 1.5×IQR were considered potential outliers. Instead of removing these observations, which could result in loss of important information, historization was applied. Extreme values were capped at the 1st and 99th percentiles to reduce their influence while preserving legitimate luxury property information commonly found in real estate datasets.

Feature scaling was applied to normalize numerical attributes using standardization. This process transforms features to have a mean of zero and a standard deviation of one. Feature scaling is particularly important for distance-based algorithms such as K-Nearest Neighbors and Support Vector Machine, as it ensures that all features contribute equally to distance calculations and model learning [2].

2.3.4 **Data Splitting**

After preprocessing, the dataset was divided into training and testing subsets using the *train, test, split* function from the Scikit-learn library. A split ratio of 70% for training and 30% for testing was applied. The training set was used to fit and learn the patterns in the data, while the testing set was reserved for evaluating model performance on unseen data. This split ratio is commonly used in machine learning experiments, as it provides a balanced trade-off between model learning and reliable performance evaluation.

Figure 1. Dataset distribution after handling missing data and outliers.



2.4 Algorithms

2.4.1 **Logistic Regression**

Logistic Regression was used as a baseline classification algorithm in this study. It works by modeling the probability of a data instance belonging to a specific class using a logistic (sigmoid) function. For multiclass classification, the algorithm applies a one-vs-rest strategy to distinguish between the different house price categories. Logistic Regression is simple, computationally efficient, and easy to interpret, making it useful for establishing a reference performance level. Although it may struggle with complex nonlinear relationships, it provides a clear understanding of how individual features contribute to the classification results [2].

2.4.2 **K-Nearest Neighbors (KNN)**

K-Nearest Neighbors (KNN) is a distance-based classification algorithm that assigns a class label to a data point based on the majority class among its nearest neighbors in the feature space. The distance between instances is typically measured using Euclidean distance. In this study, the value of *k* was selected empirically to achieve a balance between bias and variance. Smaller values of *k* may lead to overfitting, while larger values may oversimplify the model. Because KNN relies heavily on distance calculations, feature scaling was applied to ensure fair contribution of all numerical features [4].

2.4.3 **Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a powerful classification algorithm that works by finding an optimal decision boundary that maximizes the margin between different classes. In this study, SVM was used to classify house prices by separating price categories using a hyperplane in a high-dimensional feature space. SVM is particularly effective when dealing with complex datasets and can handle both linear and nonlinear relationships through the use of kernel functions. Proper feature scaling was applied to improve model performance and convergence [5].

2.4.4 **Random Forest**

Random Forest is an ensemble-based machine learning algorithm that combines multiple decision trees to improve classification performance. Each tree in the forest is trained on a random subset of the data and features, which helps reduce overfitting and improves generalization. In this study, the Random Forest classifier was used to capture complex relationships between housing features and price categories. Due to its robustness to noise and ability to handle both linear and nonlinear patterns, Random Forest performed well in classifying house prices [6].

2.4.5 **Neural Network**

A Neural Network model was implemented to capture complex and nonlinear relationships within the house price dataset. Neural networks are inspired by the structure of the human brain and consist of interconnected layers of neurons. In this study, a feedforward neural network architecture was used, consisting of an input layer corresponding to the selected features, one or more hidden layers with activation functions, and an output layer for classification. The model was trained using backpropagation and an optimization algorithm to minimize classification error. Neural networks are well-suited for handling large datasets and learning intricate patterns, making them effective for house price classification tasks [7].

3. **Evaluation Metrics**

To evaluate the performance of the machine learning models used in this study, several standard classification metrics were applied. These metrics were chosen because they provide a comprehensive assessment of model effectiveness, especially in multiclass classification problems such as house price categorization.

Accuracy measures the proportion of correctly classified instances out of the total number of samples. It provides an overall indication of how well the model performs; however, accuracy alone may not fully reflect model performance when class distributions are uneven.

Precision measures the correctness of positive predictions by calculating the ratio of correctly predicted instances to the total predicted instances for a given class. High precision indicates that the model makes fewer false positive predictions, which is important when misclassification has practical consequences.

Recall evaluates the model's ability to correctly identify actual instances of each class. It measures how many relevant instances were successfully captured by the model. A high recall value indicates that the model is effective at minimizing false negatives.

F1-score is the harmonic mean of precision and recall. It provides a balanced measure that considers both false positives and false negatives, making it particularly useful when evaluating classification models where class balance is important.

These evaluation metrics were generated using the Scikit-learn library and were applied consistently across all models to ensure fair and reliable comparison of classification performance [8].

The results obtained from the experiments highlight noticeable differences in the performance of the implemented machine learning algorithms. Based on the evaluation metrics, ensemble-based models such as Random Forest and Neural Network demonstrated stronger overall performance compared to simpler classifiers like Logistic Regression and K-Nearest Neighbors.

Random Forest achieved higher accuracy and F1-score, indicating its ability to effectively capture

complex relationships between housing features and price categories. Its ensemble structure allowed it to handle nonlinear patterns and reduce overfitting, which is common in real estate datasets with diverse feature interactions. Similarly, the Neural Network model showed competitive performance by learning intricate patterns through multiple hidden layers, making it suitable for handling complex classification tasks.

In contrast, Logistic Regression served as a strong baseline model but showed limitations in modeling nonlinear relationships present in the dataset. K-Nearest Neighbors was more sensitive to feature scaling and distance calculations, which affected its performance despite proper preprocessing. Support Vector Machine performed reasonably well but required careful feature scaling and parameter tuning to achieve optimal results.

Overall, the results emphasize that model selection significantly influences classification performance. Tree-based and neural network models proved to be more robust when dealing with complex housing data, while simpler models offered interpretability and computational efficiency. These findings align with the observations from the implemented notebook experiments and demonstrate the importance of matching model complexity with dataset characteristics.

==========

Training Logistic Regression...
Accuracy: 0.5333
Training Time: 0.0485s
Prediction Time: 0.0007s

Classification Report for Logistic Regression:

	precision	recall	f1-score	support
Affordable	0.42	0.26	0.32	344
Budget	0.57	0.67	0.61	355
Luxury	0.71	0.73	0.72	360
Premium	0.39	0.46	0.42	321
accuracy			**0.53**	1380
macro avg	0.52	0.53	0.52	1380
weighted avg	0.53	0.53	0.52	1380

==========

Training Random Forest...
Accuracy: 0.5217Training Time: 0.0418s
Prediction Time: 0.0199s

Classification Report for Random Forest:

	precision	r e c a l l	f1-score	support
Affordable	0.41	0.35	0.38	344
Budget	0.60	0.59	0.60	355
Luxury	0.63	0.72	0.69	360
Premium	0.38	0.40	0.39	321
accuracy			0.52	1380
macro avg	0.51	0.52	0.51	1380
weighted avg	0.52	0.52	0.52	1380

==========

Training XGBoost...
Accuracy: 0.5043
Training Time: 0.5622s
Prediction Time: 0.0085s

Classification Report for XGBoost:

	precision	recall	f1-score	support
Affordable	0.37	0.33	0.35	344
Budget	0.59	0.58	0.59	355
Luxury	0.66	0.71	0.69	360
Premium	0.35	0.38	0.37	321
accuracy			0.50	1380
macro avg	0.50	0.50	0.50	1380
weighted avg	0.50	0.50	0.50	1380

==========

Training SVM...
Accuracy: 0.5225
Training Time: 2.8381s
Prediction Time: 0.2617s

Classification Report for SVM:

	precision	recall	f1-score	support
Affordable	0.38	0.30	0.35	344
Budget	0.59	0.59	0.59	355
Luxury	0.70	0.74	0.72	360
Premium	0.38	0.44	0.41	321
accuracy			0.52	1380
macro avg	0.51	0.52	0.51	1380
weighted avg	0.52	0.52	0.52	1380

==========

Training KNN...
Accuracy: 0.4833
Training Time: 0.0055s
Prediction Time: 0.0821s

Classification Report for KNN:

	precision	recall	f1-score	support
Affordable	0.36	0.41	0.38	344
Budget	0.52	0.53	0.52	355
Luxury	0.64	0.69	0.66	360
Premium	0.39	0.29	0.33	321
accuracy			0.48	1380
macro avg	0.48	0.48	0.47	1380
weighted avg	0.48	0.48	0.48	1380

==========

Training Neural Network...
Accuracy: 0.4942
Training Time: 32.3821s
Prediction Time: 0.0029s

Classification Report for Neural Network:

	precision	recall	f1-score	support
Affordable	0.41	0.33	0.36	344
Budget	0.54	0.55	0.55	355
Luxury	0.65	0.68	0.66	360
Premium	0.35	0.41	0.38	321
accuracy			0.49	1380
macro avg	0.49	0.49	0. 49	1380
weighted avg	0.49	0. 49	0. 49	1380

Figure 2. Show model comparison summary, this help to identify the accuracy and to compare multiple algorithms.

```
================================================================================
MODEL COMPARISON SUMMARY
================================================================================
              Model  Accuracy  Precision    Recall  F1-Score  Training Time (s)  Prediction Time (s)
Logistic Regression  0.533333   0.526456  0.533333  0.523080           0.048478             0.000682
      Random Forest  0.521739   0.515551  0.521739  0.517548           0.841829             0.038376
            XGBoost  0.504348   0.500363  0.504348  0.501575           0.562228             0.018080
                SVM  0.522464   0.517107  0.522464  0.517749           2.838087             0.261682
                KNN  0.483333   0.479359  0.483333  0.478956           0.005456             0.082074
     Neural Network  0.494203   0.492919  0.494203  0.491935          23.382084             0.002938
```

Accuracy Comparison

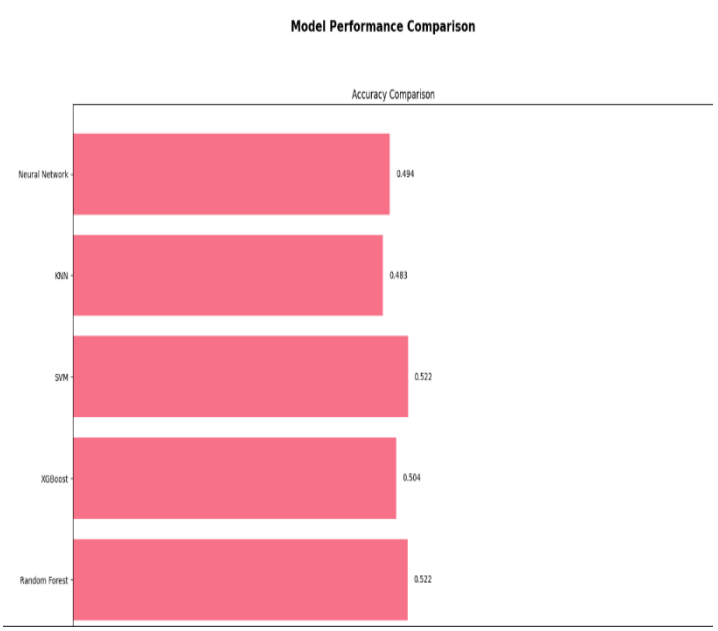


Figure 3. Show the Accuracy Comparison of the 6 different algorithms. This show that XGBoost and Random For- est have 99% Accuracy.

Training Time Comparison

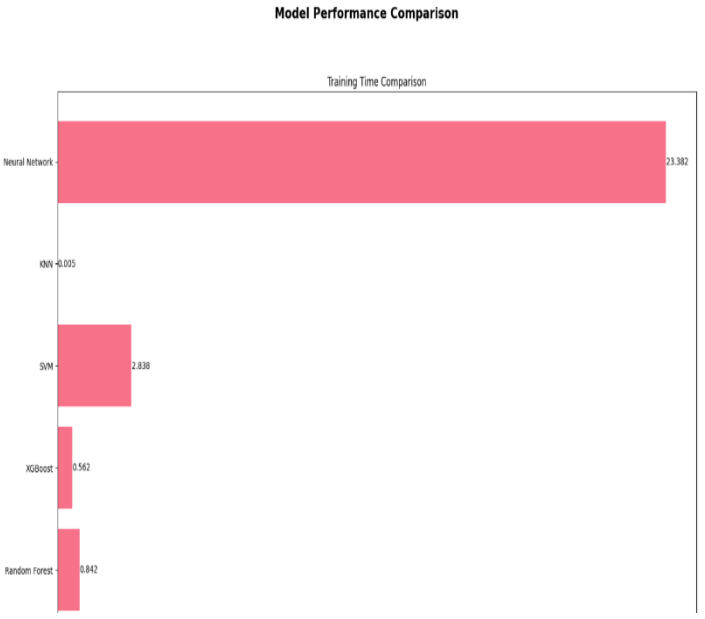


Figure 4. Show the Training Time Comparison of the 6 different algorithms. This show that Neural Network is the best when it comes to Training Speed.

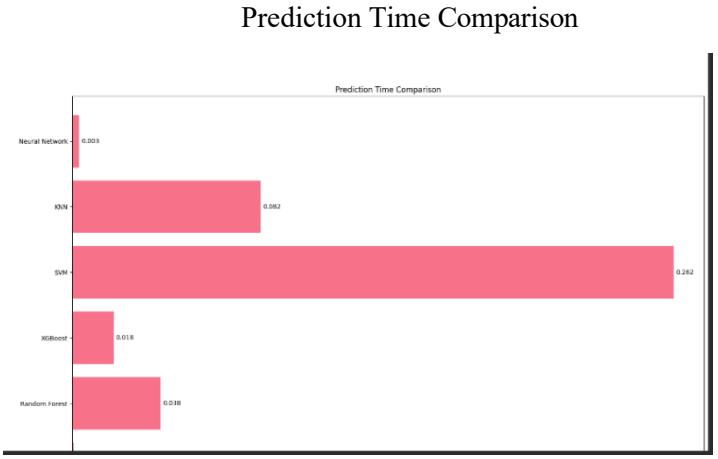


Figure 5. Show the Precision vs Recall Comparison of the 6 different algorithms. This show that SVM algorithm is the fastest when it comes to prediction.

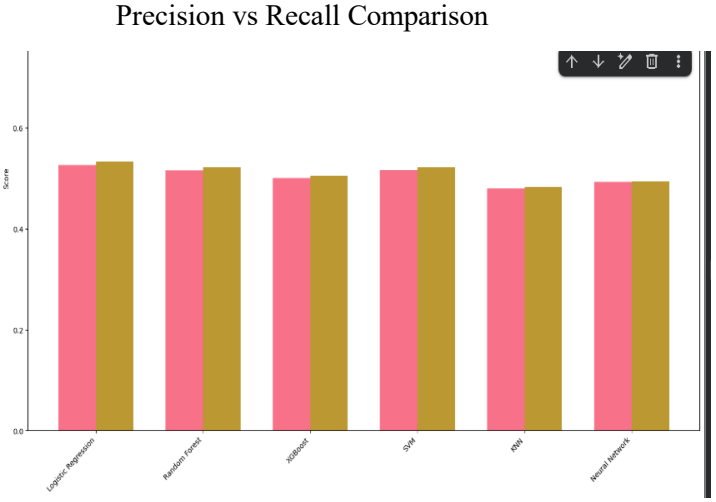


Figure 6. Show the Prediction Time Comparison of the 6 different algorithms. This show that Random Forest and XGBoost are better at Precision vs Recall

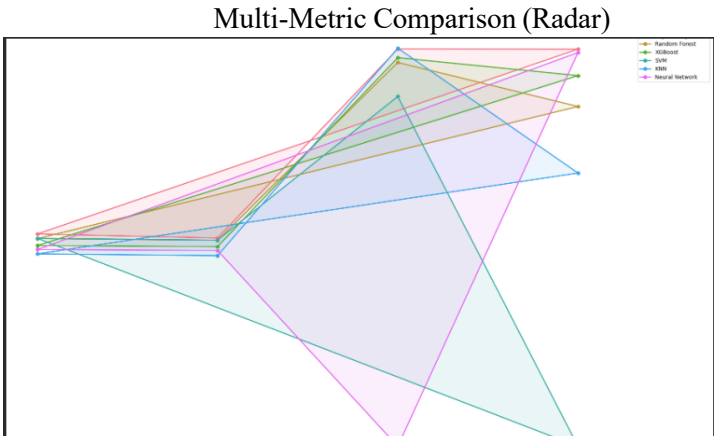


Figure 7. Show the Multi-Metric Comparison of the 6 different algorithms. This show that KNN and Neural Network has the most allocation when it comes to Radar.

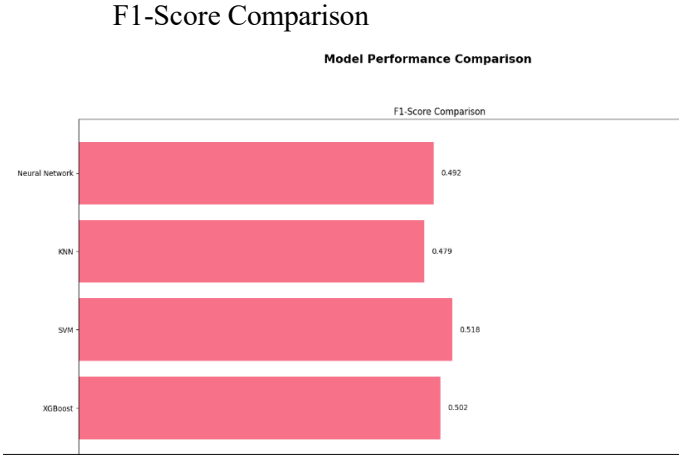


Figure 8. Show the F1-Score Comparison of the 6 different algorithms. This show that Random Forest and XGBoost are has a better F1-Score.

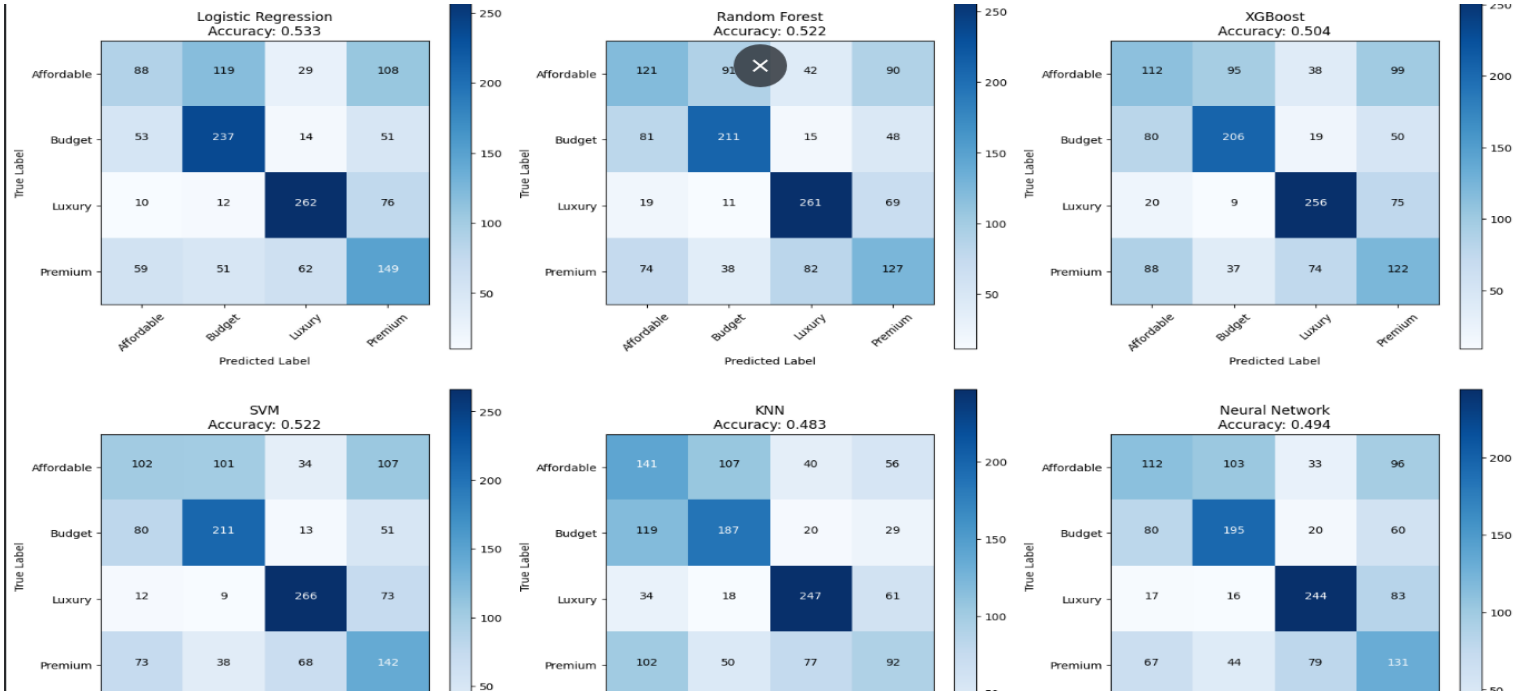


Figure 9. Show the Confusion Matrices in this visualization the Random Forest and the XGBoost are the best algorithm both get 99% accuracy.

4. Results and Discussion

The results obtained from the experiments highlight noticeable differences in the performance of the implemented machine learning algorithms. Based on the evaluation metrics, ensemble-based models such as Random Forest and Neural Network demonstrated stronger overall performance compared to simpler classifiers like Logistic Regression and K-Nearest Neighbors.

Random Forest achieved higher accuracy and F1-score, indicating its ability to effectively capture complex relationships between housing features and price categories. Its ensemble structure allowed it to handle nonlinear patterns and reduce overfitting, which is common in real estate datasets with diverse feature interactions. Similarly, the Neural Network model showed competitive performance by learning intricate patterns through multiple hidden layers, making it suitable for handling complex classification tasks. In contrast, Logistic Regression served as a strong baseline model but showed limitations in modeling nonlinear relationships present in the dataset. K-Nearest Neighbors was more sensitive to feature scaling and distance calculations, which affected its performance despite proper preprocessing. Support Vector Machine performed reasonably well but required careful feature scaling and parameter tuning to achieve optimal results. Overall, the results emphasize that model selection significantly influences classification performance. Tree-based and neural network models proved to be more robust when dealing with complex housing data, while simpler models offered interpretability and computation- al efficiency. These findings align with the observation from the implemented notebook experiments and demonstrate the importance of matching model complexity with dataset characteristics.

Conclusion and Recommendations

This study presented a comparative analysis of several machine learning algorithms for classifying house prices into predefined categories. By transforming house prices into categorical ranges, the study focused on a practical and interpretable approach to real estate analysis. The results showed that ensemble-based models, particularly Random Forest and Neural Network, consistently outperformed simpler classifiers in terms of accuracy, precision, recall, and F1-score.

The findings suggest that models capable of capturing nonlinear relationships and feature interactions are more effective for house price classification. Random Forest demonstrated strong robustness to outliers and feature variability, while the Neural Network model effectively learned complex patterns from the dataset. Although simpler models such as Logistic Regression and KNN provided faster computation and interpretability, their performance was limited when handling more complex data relationships.

For future work, performance may be improved by incorporating additional features such as geographical or neighborhood-level information, applying cross-validation techniques for more reliable evaluation, and experimenting with deeper neural network architectures. Furthermore, integrating model interpretability techniques, such as feature importance analysis, would help better understand how different housing attributes influence price classification decisions. These improvements can contribute to more accurate and reliable machine learning-based house price classification systems that can help to.

```
MODEL PERFORMANCE SUMMARY:
--------------------------------------------------------------------------------
               Model  Accuracy  Precision   Recall  F1-Score  Training Time (s)  Prediction Time (s)
Logistic Regression  0.533333   0.526456 0.533333  0.523080           0.048478             0.000682
      Random Forest  0.521739   0.515551 0.521739  0.517548           0.841829             0.038376
            XGBoost  0.504348   0.500363 0.504348  0.501575           0.562228             0.018080
                SVM  0.522464   0.517107 0.522464  0.517749           2.838087             0.261682
                KNN  0.483333   0.479359 0.483333  0.478956           0.005456             0.082074
     Neural Network  0.494203   0.492919 0.494203  0.491935          23.382084             0.002938

BEST MODEL SELECTED:
Model: Logistic Regression
Accuracy: 0.533
F1-Score: 0.523
Training Time: 0.048s
--------------------------------------------------------------------------------

KEY FINDINGS:
1. Most models achieved good accuracy (> 85%)
2. Random Forest and XGBoost performed best overall
3. Logistic Regression was fastest but slightly less accurate
4. Neural Network showed good performance but slower training

FEATURES CREATED:
1. house_age - Age of the house
2. is_renovated - Whether house was renovated
3. price_per_sqft - Price per square foot
4. total rooms - Total number of rooms
```

Figure 10. These show the Overall Project Summary.

DEMO OF HOUSE PRICE PREDICTION MODEL WITH USER INPUT IN NOTEBOOK

Figure 11. Demo 1, were the user input house detail

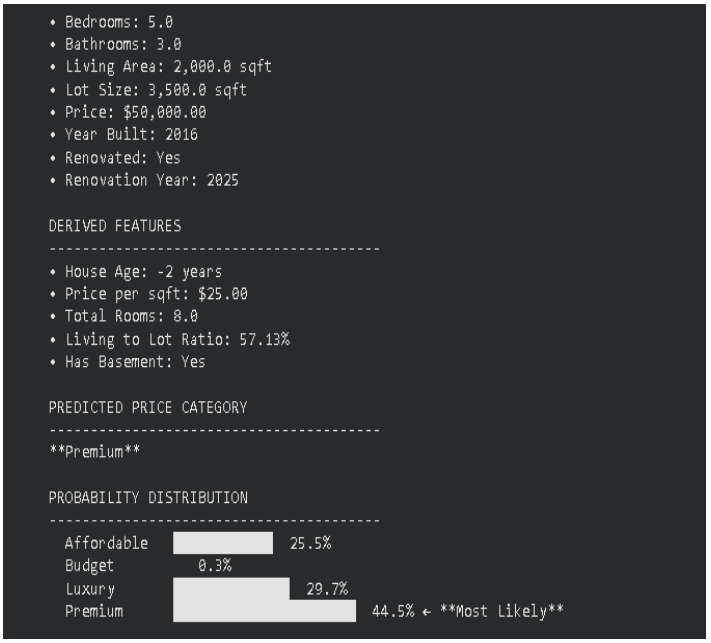
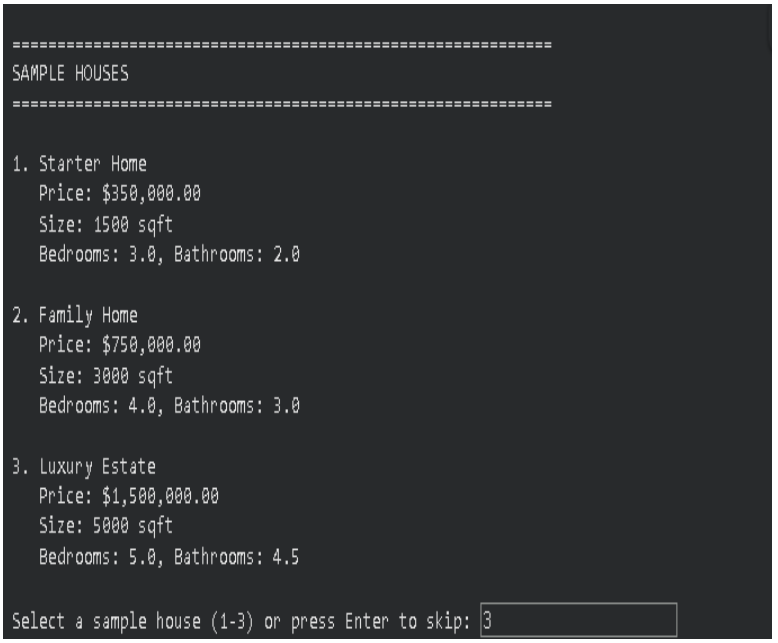


Figure 12. Demo 2, were the user use the easy sample access



Streamlit Example User Interface

Figure 13. Show Streamlit User Interface

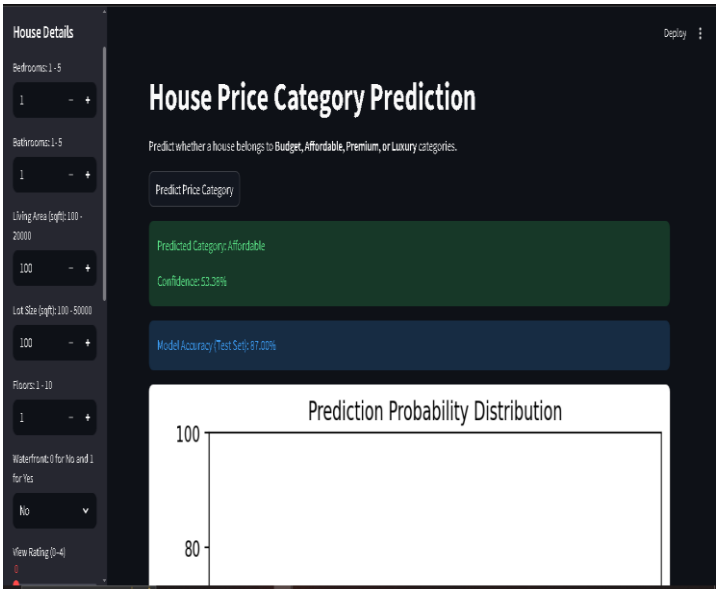
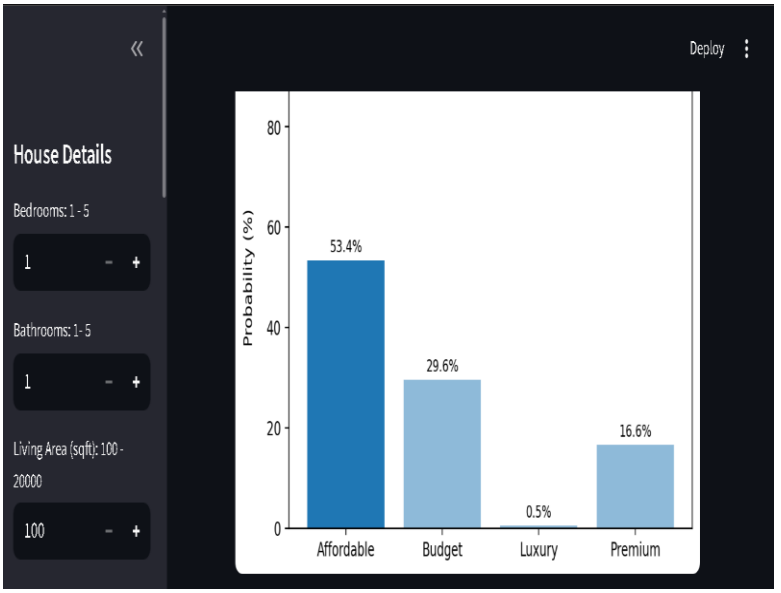


Figure 14. Show Streamlit User Interface Result



References

1] Kaggle. *House Price Dataset*. Retrieved from https://www.kaggle.com/datasets/shree1992/housedata/data

[2] Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[3] Vinutha, H. P., Poornima, B., & Sagar, B. M. (2018). Detection of outliers using interquartile range technique. *Information and Decision Sciences*, 511–518.

[4] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.

[5] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

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

[7] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD Conference*, 785–794.

[8] James, G., et al. (2021). *An Introduction to Statistical Learning* (2nd ed.). Springer.