Rental Price Tier Classification

Using Machine Learning

Data Mining CISC-5790

Team Members:

Akash Kumar Yadav

Aryan Varmora

Yunshuo Kyle Tian

Fordham University, Lincoln Center

113 W 60th Street, New York, NY 10023, USA

Instructor: Prof. Yijun Zhao

April 20, 2025

ABSTRACT:

This project analyzes a substantial dataset comprising around 100,000 apartment listings across the United States, sourced from the University of California, Irvine (UCI). The dataset encapsulates a diverse array of attributes, including pricing, amenities, and geographical location, which are pivotal in understanding the dynamics of the apartment rental market. The primary objective of this research is to identify and analyze trends in rental pricing, examine how geographical location influences apartment features and prices, and determine the impact of amenities on rental costs. The study uses multivariate analysis techniques to explore relationships within the data, facilitating the development of predictive models and innovative feature engineering, such as calculating bed per square foot and categorizing listings into luxury versus budget segments.

Additionally, an interactive map that we created using Python will allow users to see listings dynamically and visualize real-time market prices across different states in the United States of America. This report aims to deliver actionable insights that could benefit renters, real estate professionals, and policymakers, enhancing understanding of the complex factors that drive the US apartment rental market.

INTRODUCTION:

In today's data-driven environment, analyzing, predicting, and displaying complex datasets is critical for gaining insights, predicting prices, and making sound decisions. This project uses advanced Machine Learning Algorithms to analyze and predict the price category of a real-world dataset of US apartment listings obtained from the University of California, Irvine (UCI). The purpose is to discover rental market factors across states and localities, such as price prediction, property availability, and the impact of variables such as pet policies and bedroom counts.

EXPLORATORY DATA ANALYSIS (EDA):

Data Description:
The data utilized in this project is sourced from the University of California, Irvine(UCI), and the dataset used in this analysis comprises 99,105 apartment listings across the United States. This dataset is a rich repository of both numerical and categorical data, providing a comprehensive snapshot of the current U.S. apartment rental market. The dataset includes 23 critical features such as pricing, amenities, the number of bedrooms and bathrooms, square footage, and geographical information (latitude, longitude, city, and state).

Total Observations: 99,492
• **Total Features:** 22

• **Feature Types: –**

- **Numerical Features:** bedrooms, bathrooms, square_feet, latitude, longitude, amenities
- **Categorical Features:** state, pet_allowed, price_type, has_photo, cityname
- **Target Variable:** price_category(High, Medium, Low)

```
          id                category                           title  \
0  5668640009  housing/rent/apartment       One BR 507 & 509 Esplanade
1  5668639818  housing/rent/apartment      Three BR 146 Lochview Drive
2  5668639686  housing/rent/apartment  Three BR 3101 Morningside Drive
3  5668639659  housing/rent/apartment            Two BR 209 Aegean Way
4  5668639374  housing/rent/apartment         One BR 4805 Marquette NE

                                        body amenities  bathrooms  \
0  This unit is located at 507 & 509 Esplanade, R...       NaN        1.0
1  This unit is located at 146 Lochview Drive, Ne...       NaN        1.5
2  This unit is located at 3101 Morningside Drive...       NaN        2.0
3  This unit is located at 209 Aegean Way, Vacavi...       NaN        1.0
4  This unit is located at 4805 Marquette NE, Alb...       NaN        1.0

   bedrooms currency fee  has_photo  ... price_display  price_type  \
0       1.0      USD  No  Thumbnail  ...        $2,195     Monthly
1       3.0      USD  No  Thumbnail  ...        $1,250     Monthly
2       3.0      USD  No  Thumbnail  ...        $1,395     Monthly
3       2.0      USD  No  Thumbnail  ...        $1,600     Monthly
4       1.0      USD  No  Thumbnail  ...          $975     Monthly

  square_feet             address       cityname state latitude longitude  \
0         542   507  509 Esplanade  Redondo Beach    CA  33.8520 -118.3759
1        1500      146 Lochview Dr   Newport News    VA  37.0867  -76.4941
2        1650  3101 Morningside Dr        Raleigh    NC  35.8230  -78.6438
...
 21  time          99492 non-null  int64
dtypes: float64(5), int64(3), object(14)
```

Figure 1: Original dataset details

Dataset Criteria Satisfaction:

• Multivariate Analysis Capability: The dataset supports complex multivariate analyses due to its diverse range of numerical and categorical variables, making it ideal for predictive modeling and exploratory data analysis.

• Public Accessibility: As a dataset curated by a reputable educational institution (UCI), it is publicly available and meets the project requirements for using non-classified, accessible data.

• Feature Engineering Potential: The dataset allows for extensive feature engineering, such as calculating price per square foot or categorizing properties into luxury versus budget, enhancing the depth of the analysis.

Dependent and Independent Variables:

• Dependent Variable: The primary dependent variable in this dataset is price_display, representing the rental price of each apartment. Most analyses will aim to predict or explain this variable of interest.

• Independent Variables: Independent variables include amenities, bathrooms, bedrooms, square_feet, state, and cityname, among others. These variables are expected to influence the rental pricing and are used to understand variations in apartment costs across different markets.

Numerical Feature Distributions:

The following figure shows the distribution of numerical features: bedrooms, bathrooms, square_feet, latitude, longitude, and amenities. These visualizations help us understand the spread, central tendency, and possible anomalies (e.g., zeros and outliers) in the numerical data.

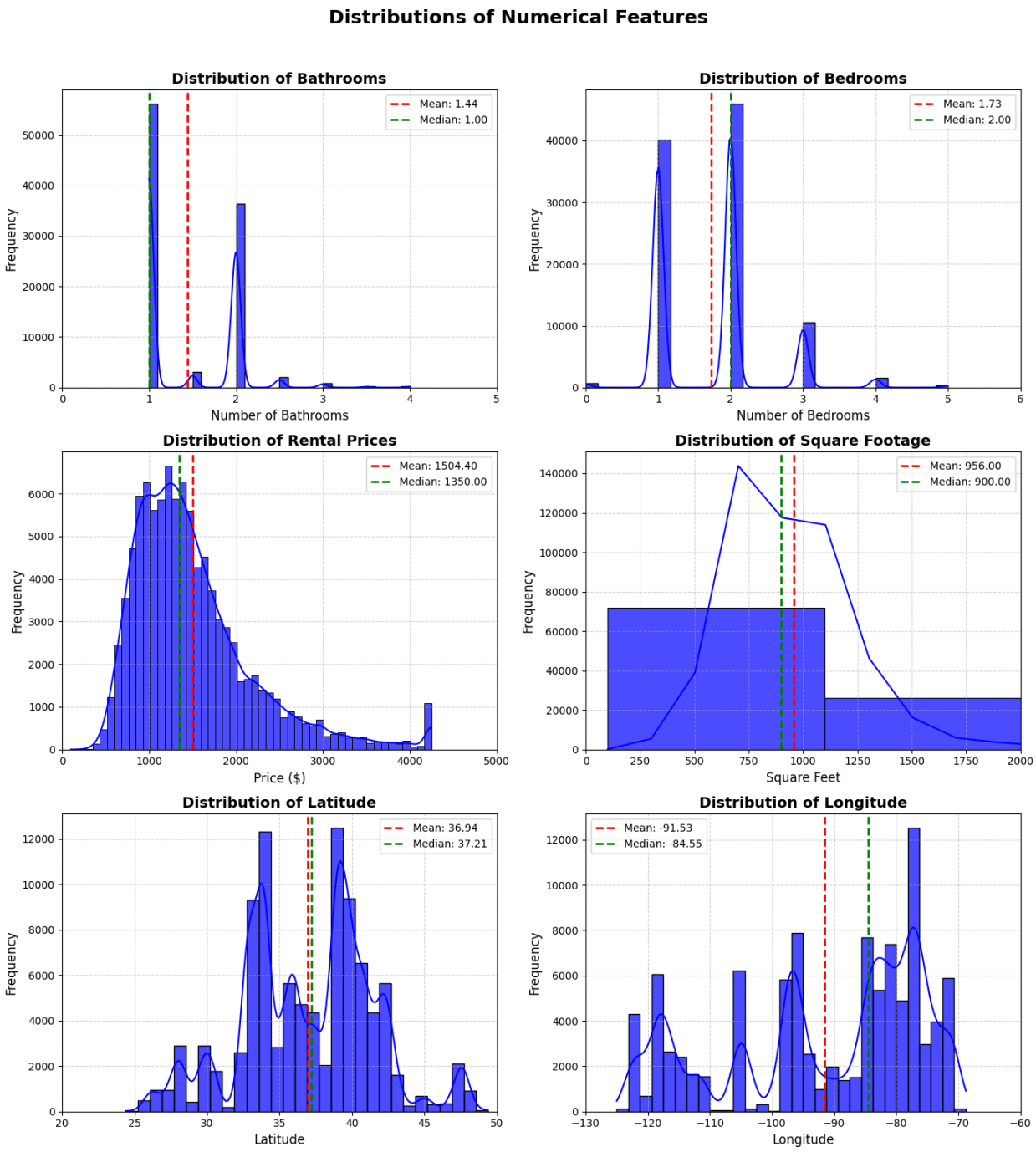


Figure 2: Distributions of Numerical Features

Observations:

- ★ **Bathrooms:** The distribution is dominated by 1-bathroom listings, followed by a significant count of 2-bathroom listings.
- ★ **Bedrooms:** Listings primarily feature 1 or 2 bedrooms, with the highest frequency at two bedrooms.
- ★ **Rental Prices:** Price distribution is right-skewed, with most listings priced below $2000 and a median of $1350.
- ★ **Square Footage:** Square footage distribution is also right-skewed, predominantly under 1250 sq ft, with a median of 900 sq ft.
- ★ **Latitude:** Latitude distribution exhibits multiple peaks, suggesting geographic clustering, notably around 34-35 and 39-40 degrees.
- ★ **Longitude:** Longitude distribution is distinctly multimodal, reflecting several geographically separate cluster listings across the US.

Target Variable Distribution:

The Target variable, Price, has been categorized into High, Medium, and Low rental prices in the dataset. Understanding its distribution is crucial for evaluating class imbalance, which can impact the performance of our models.

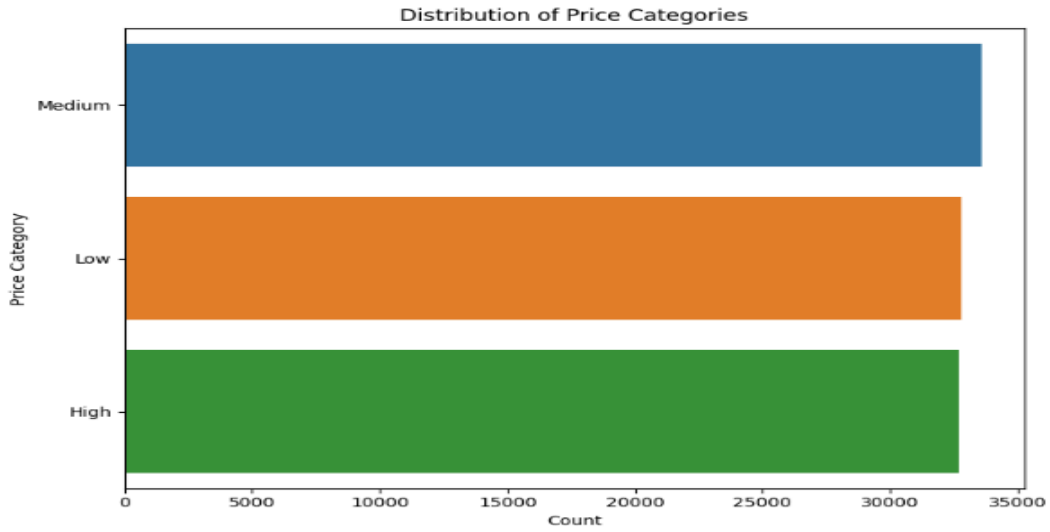


Figure 3: Distribution of Price Categories

From Figure 3, we observe the following:

The dataset is relatively balanced across the 'Low', 'Medium', and 'High' price categories. 'Low' and 'Medium' categories have nearly identical high counts, while the 'High' category shows a slightly lower but still substantial count. The listing was almost equally distributed across the 'Low' (33.33%), 'Medium' (33.34%), and 'High' (33.33%) price categories.

Feature-Target Relationships:

Understanding the relationship between features and the target variable (price_category) is critical to uncovering patterns and dependencies that help improve the model's accuracy in prediction. Below, we analyze the association between numerical features and the target variable.

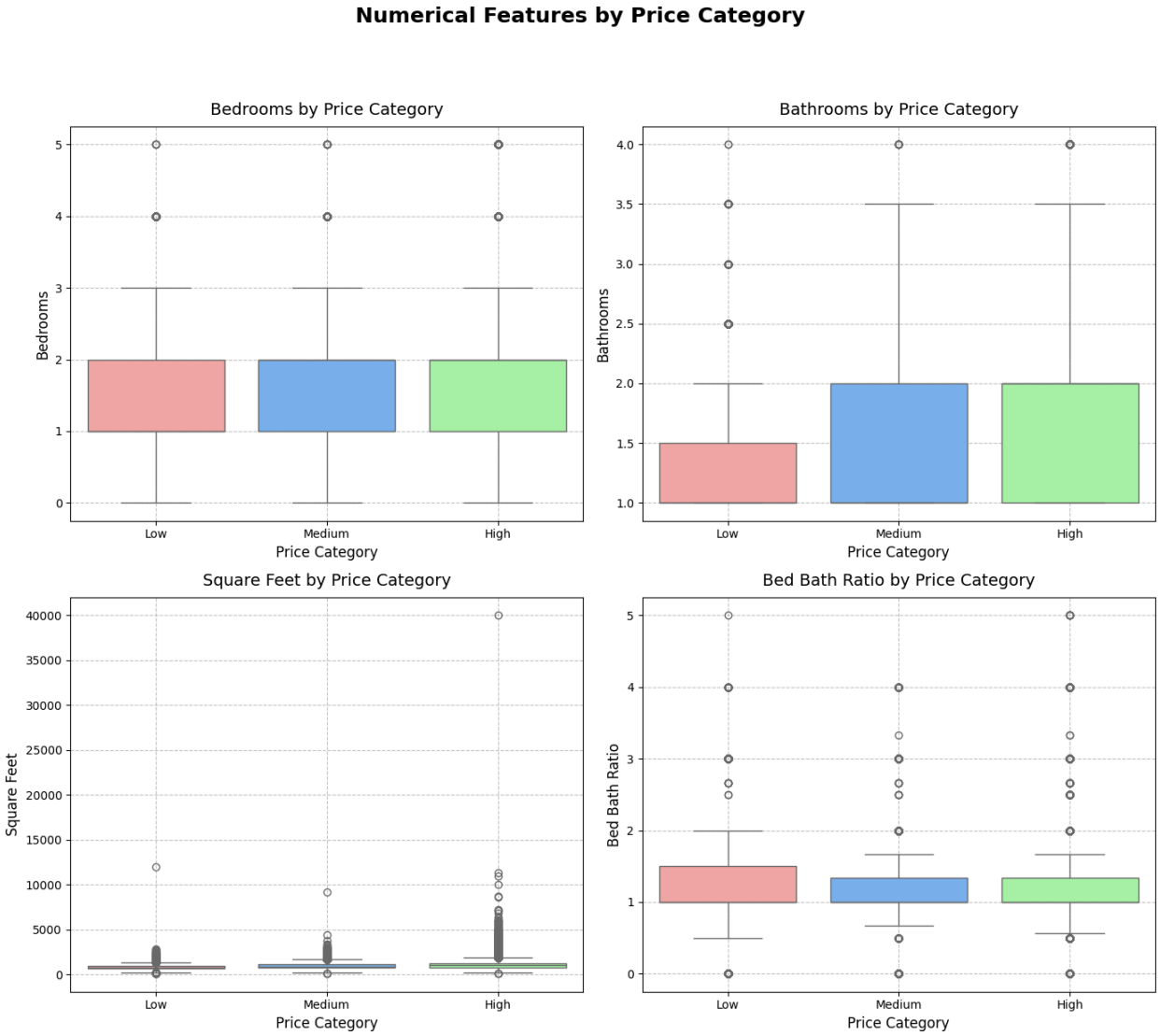


Figure 4: Boxplots of Numerical Features by Target Class (price_category)

Observation:

1. The Median bedrooms increase slightly from Low to Medium and High, with distributions overlapping significantly for medium and high prices.
2. Higher price categories clearly correlate with an increasing median number of bathrooms and a wider range of values.
3. Square feet strongly increases in both median value and variability as the price category moves from Low to High.

4. The median ratio of bedrooms to bathrooms is relatively stable across categories, but higher-priced listings show much greater variability in this ratio.
5. There are several outliers observed in square feet, Bed Bath ratio, and square feet per bedroom, which will require careful handling during preprocessing.

CORRELATION ANALYSIS:

Numerical Feature Correlations:

We have analyzed the correlation matrix using the Pearson correlation coefficient to identify relationships among numerical features.

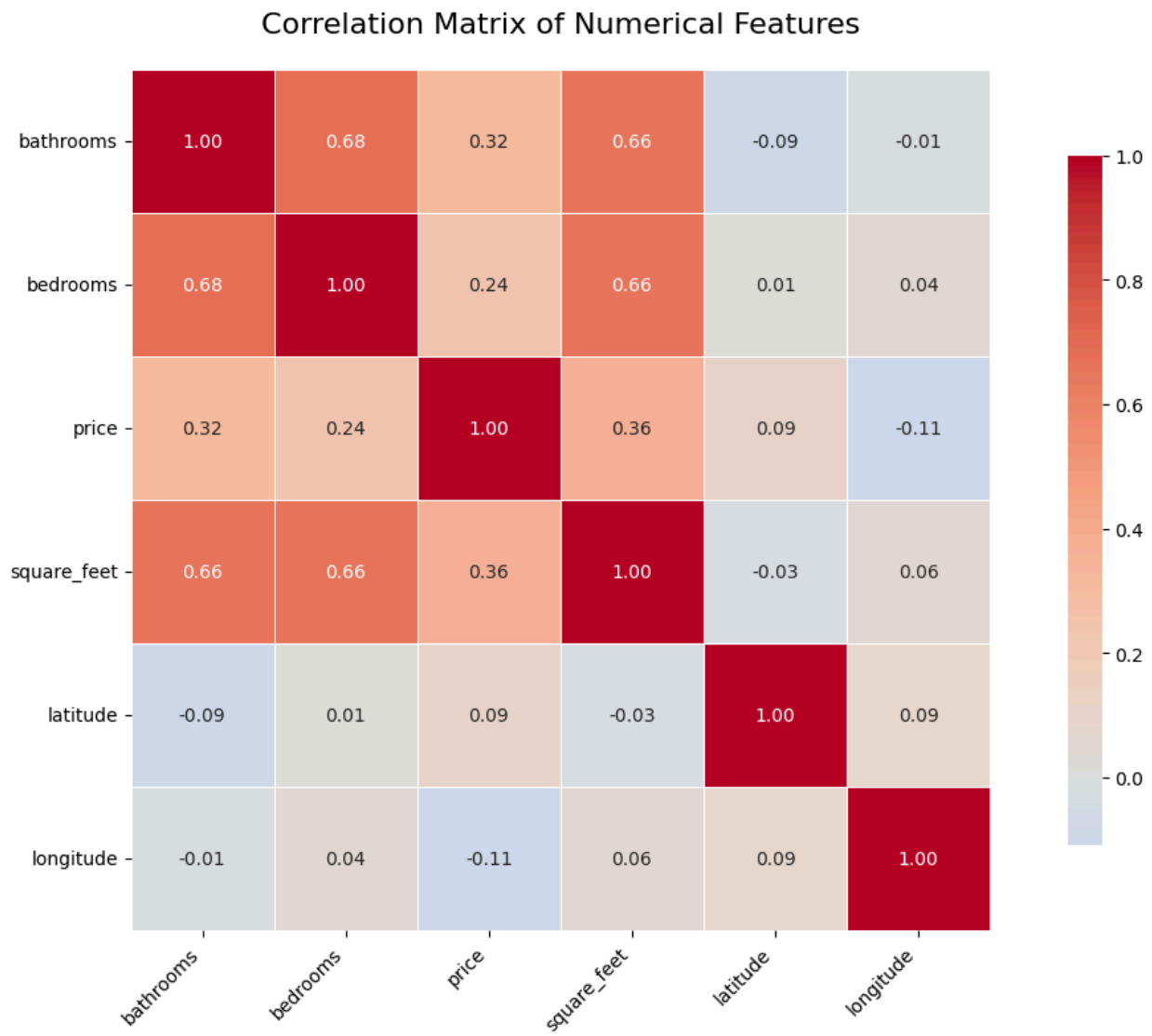


Figure 5: Correlation Matrix of Numerical Features

Key observations:

- ❖ There are high positive correlations (0.66-0.68) among bathrooms, bedrooms, and square feet, indicating multicollinearity among the size features.
- ❖ square_feet (0.36) and bathrooms (0.32) exhibit the highest, although moderate, positive correlations with price among the remaining features like bedrooms (0.24).
- ❖ The geographic features (latitude, longitude) have extremely weak linear correlations (close to zero) with price and the size features (bathrooms, bedrooms, square_feet).

Categorical Feature Correlations:

We have analyzed categorical features correlation using Cramer's V correlation, as shown in Figure 6.

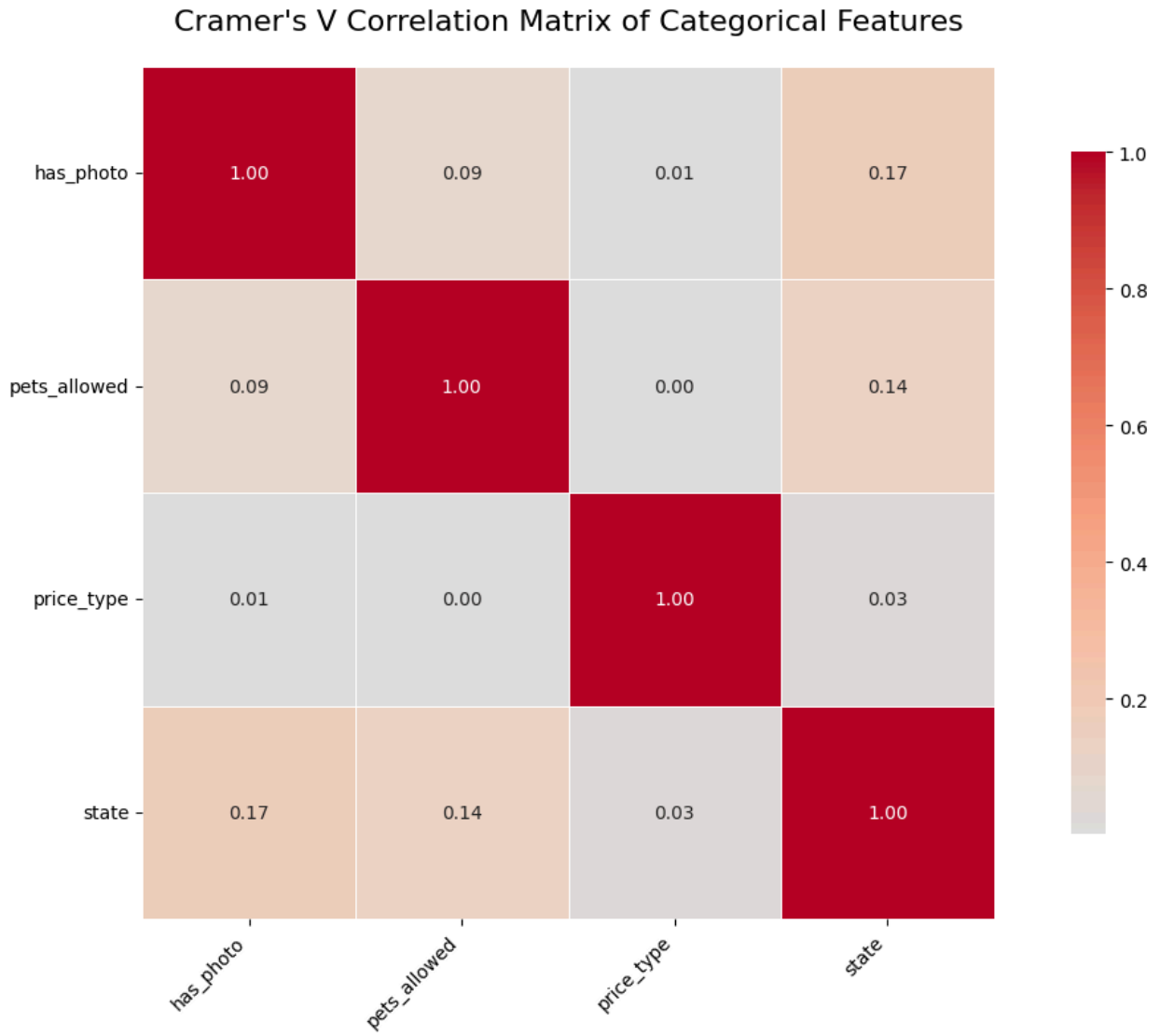


Figure 6: Correlation Matrix of Categorical Features

Key observations:

- ❖ In general, the relationships among these categorical variables such as has_photo, pets_allowed, price_type, state are fairly weak, as indicated by all off-diagonal Cramer's V statistics below 0.2.

- ❖ The strongest relationships are between state and has_photo(0.17) and between state and pets_allowed(0.14), suggesting some geographic variation in photo availability and pet acceptance.

Correlation Between Features and Target Variable:
We have analyzed the relationship between features and the target variable (price_category), we visualized featuretarget correlations for both numerical and categorical features.

DATA PRE-PROCESSING:

In preparing the dataset for analysis, I implemented several data-cleaning strategies to handle missing values, ensuring both the integrity and usability of the data. Each method was carefully chosen based on the specific characteristics of the data and the analytical requirements of the project:

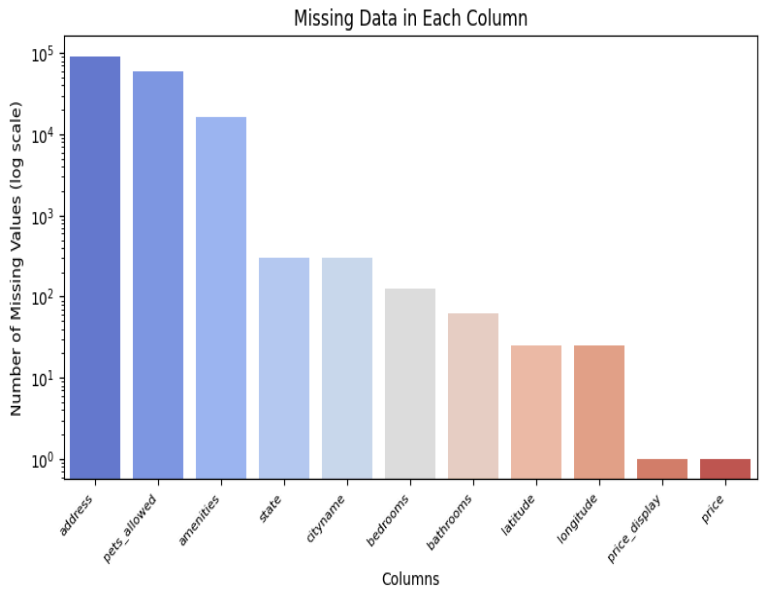


Figure 7: Graph for missing values

Missing Data Summary	
Column Name	Missing Values
address	91549
pets_allowed	60424
amenities	16044
state	302
cityname	302
bedrooms	124
bathrooms	63
latitude	25
longitude	25
price_display	1
price	1

Figure 8: Number of missing values

1. Filling Missing Values with Default Text:
I filled in missing values for the **amenities** column with 'Not Specified'. This approach allowed me to preserve the dataset's structure while marking the absence of detailed amenities, which might influence the analytical results on how amenities impact rental prices.

In the **pets_allowed** column, I filled in missing entries with 'Unknown'. To maintain consistency within the dataset, I further refined this by ensuring that all entries align with expected categories ("None", "Cats", "Dogs", "Cats, Dogs"). Any deviations were also categorized as 'Unknown'. This standardization is essential for analyzing the impact of pet policies on the rental market.

Missing values in **cityname** and state were filled with 'Unknown', allowing the retention of as many entries as possible for broader analyses while acknowledging the data's incompleteness.

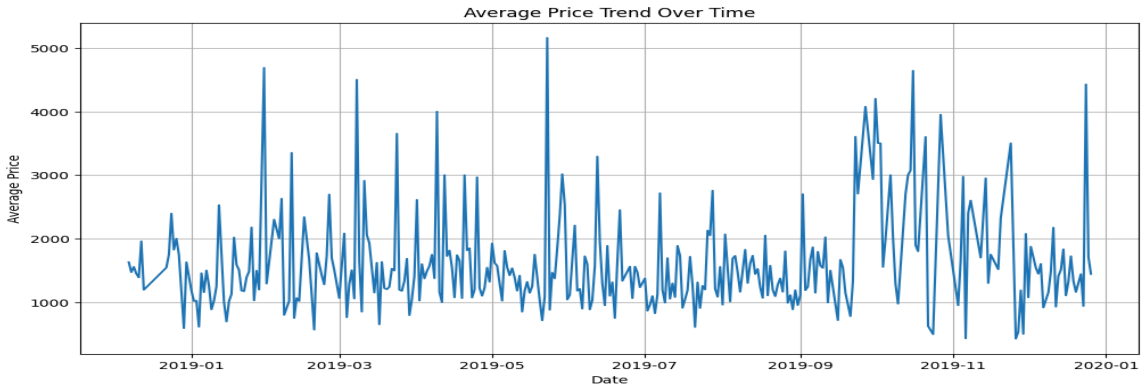
2. Replacing Missing Values with Median:
Missing values in the bedrooms and bathroom columns were replaced with their respective medians. This technique is particularly suitable for handling numerical data where preserving the central tendency is crucial. The median, being robust against outliers, ensures that the replacement does not distort the data distribution, thus maintaining the integrity of the dataset.

3. Dropping Rows with Missing Critical Information:
I removed rows missing critical information, such as address and price_display. Given the centrality of these variables to geographic and price analyses, their absence would significantly impair the reliability of subsequent insights.

4. Dropping Unimportant Column:
Due to a high proportion of missing entries, I opted to drop the address column entirely. This decision was based on the judgment that the address details would not significantly impact broad market trend analyses and would simplify the dataset by eliminating a variable with extensive incompleteness.

By employing various cleaning techniques, from median imputation to eliminating rows or columns, I ensured that the dataset was tailored for specific analytical needs and enhanced for reliability without compromising data validity. This careful preparation facilitates robust and insightful real estate market analysis and prediction.

5. Adding a substantial column:
We added a new column, datetime, to the dataset, which is valuable for data visualization and price prediction in the future. Here is a Figure 9, where you can visualize an average price trend over time.



Cleaned Dataset:

	id	category	title	body	amenities	bathrooms	bedrooms	currency	fee	has_photo	...	price_type	square_feet	cityname	state	latitude	longitude	source	time	state_name	datetime
0	5668640009	housing/rent/apartment	One BR 507 & 509 Esplanade	This unit is located at 507 & 509 Esplanade, R...	Not Specified	1.0	1.0	USD	No	Thumbnail	...	Monthly	542	Redondo Beach	CA	33.8520	-118.3759	RentLingo	1577360355	California	2019-12-26 11:39:15
1	5668639818	housing/rent/apartment	Three BR 146 Lochview Drive	This unit is located at 146 Lochview Drive, Ne...	Not Specified	1.5	3.0	USD	No	Thumbnail	...	Monthly	1500	Newport News	VA	37.0867	-76.4941	RentLingo	1577360340	Virginia	2019-12-26 11:39:00
2	5668639686	housing/rent/apartment	Three BR 3101 Morningside Drive	This unit is located at 3101 Morningside Drive...	Not Specified	2.0	3.0	USD	No	Thumbnail	...	Monthly	1650	Raleigh	NC	35.8230	-78.6438	RentLingo	1577360332	North Carolina	2019-12-26 11:38:52
3	5668639659	housing/rent/apartment	Two BR 209 Aegean Way	This unit is located at 209 Aegean Way, Vacavi...	Not Specified	1.0	2.0	USD	No	Thumbnail	...	Monthly	820	Vacaville	CA	38.3622	-121.9712	RentLingo	1577360330	California	2019-12-26 11:38:50
4	5668639374	housing/rent/apartment	One BR 4805 Marquette NE	This unit is located at 4805 Marquette NE, Alb...	Not Specified	1.0	1.0	USD	No	Thumbnail	...	Monthly	624	Albuquerque	NM	35.1038	-106.6110	RentLingo	1577360308	New Mexico	2019-12-26 11:38:28

[5 rows x 23 columns]

Figure 4: Snapshot of the cleaned dataset

	id	bathrooms	bedrooms	price	square_feet	latitude	longitude	time	datetime
count	9.910500e+04	99105.000000	99105.000000	99105.000000	99105.000000	99105.000000	99105.000000	9.910500e+04	99105
mean	5.358114e+09	1.445149	1.728682	1525.507694	955.996045	36.941309	-91.555552	1.559652e+09	2019-06-04 12:39:48.780596736
min	5.121046e+09	1.000000	0.000000	100.000000	101.000000	19.573800	-159.369800	1.544174e+09	2018-12-07 09:20:18
25%	5.197948e+09	1.000000	1.000000	1012.000000	729.000000	33.743600	-104.817100	1.550832e+09	2019-02-22 10:33:41
50%	5.508672e+09	1.000000	2.000000	1350.000000	900.000000	37.213900	-84.549400	1.568745e+09	2019-09-17 18:29:48
75%	5.509006e+09	2.000000	2.000000	1795.000000	1115.000000	39.955900	-77.576700	1.568767e+09	2019-09-18 00:35:46
max	5.669439e+09	9.000000	9.000000	52500.000000	40000.000000	64.833200	-68.778800	1.577391e+09	2019-12-26 20:17:05
std	1.846979e+08	0.547074	0.748731	903.660869	387.732138	4.604559	15.832541	1.105025e+07	NaN

Figure 5: Statistical Summary of Numerical Features

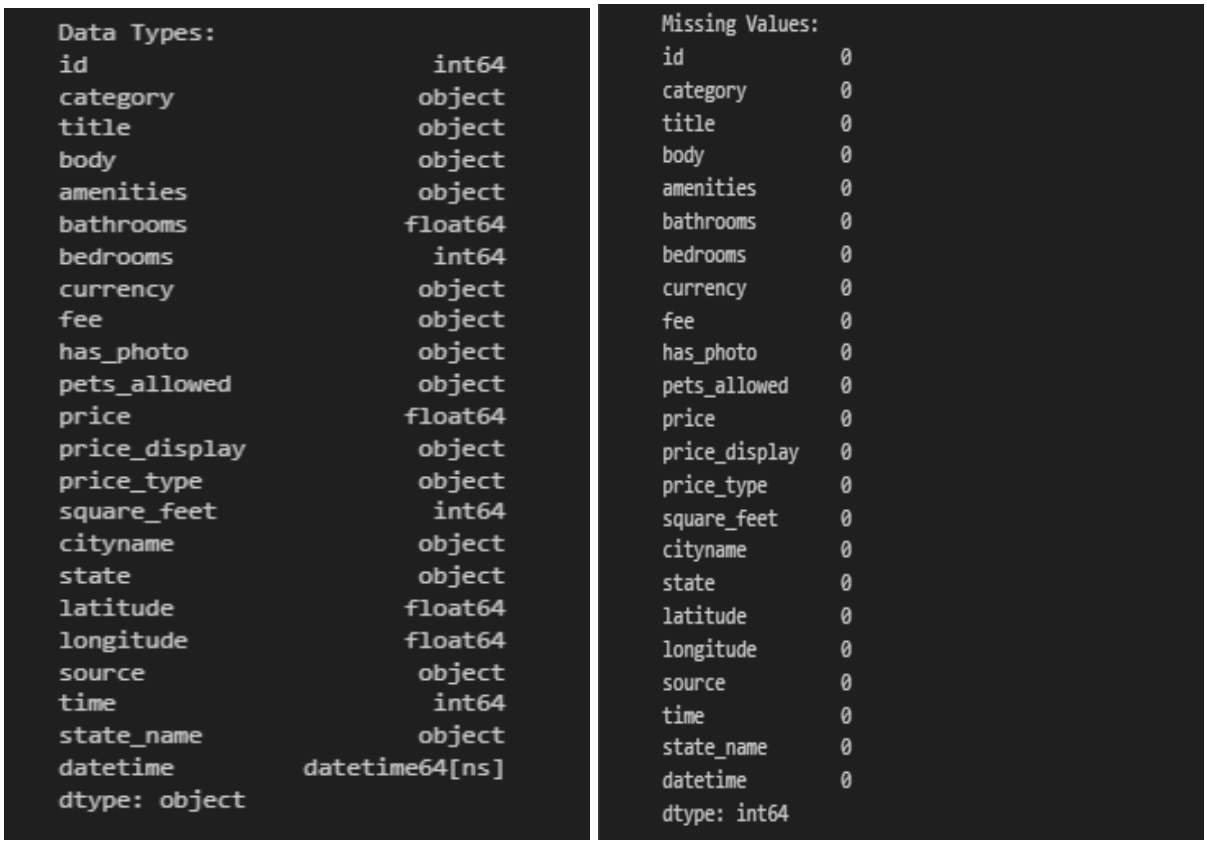


Figure 8: Data Types and Missing Values in the cleaned Dataset

Dataset Splitting:

The dataset was split into training and testing sets using an 80-20 split after handling invalid points, missing values, and invalid datatypes to ensure no erroneous values influenced the split. Other preprocessing steps were performed after splitting the dataset to avoid data leakage.

Outlier Detection:

Outliers in numerical features were identified using the Interquartile Range (IQR) method: Lower Bound = Q1 − 1.5 · IQR, Upper Bound = Q3 + 1.5 · IQR.

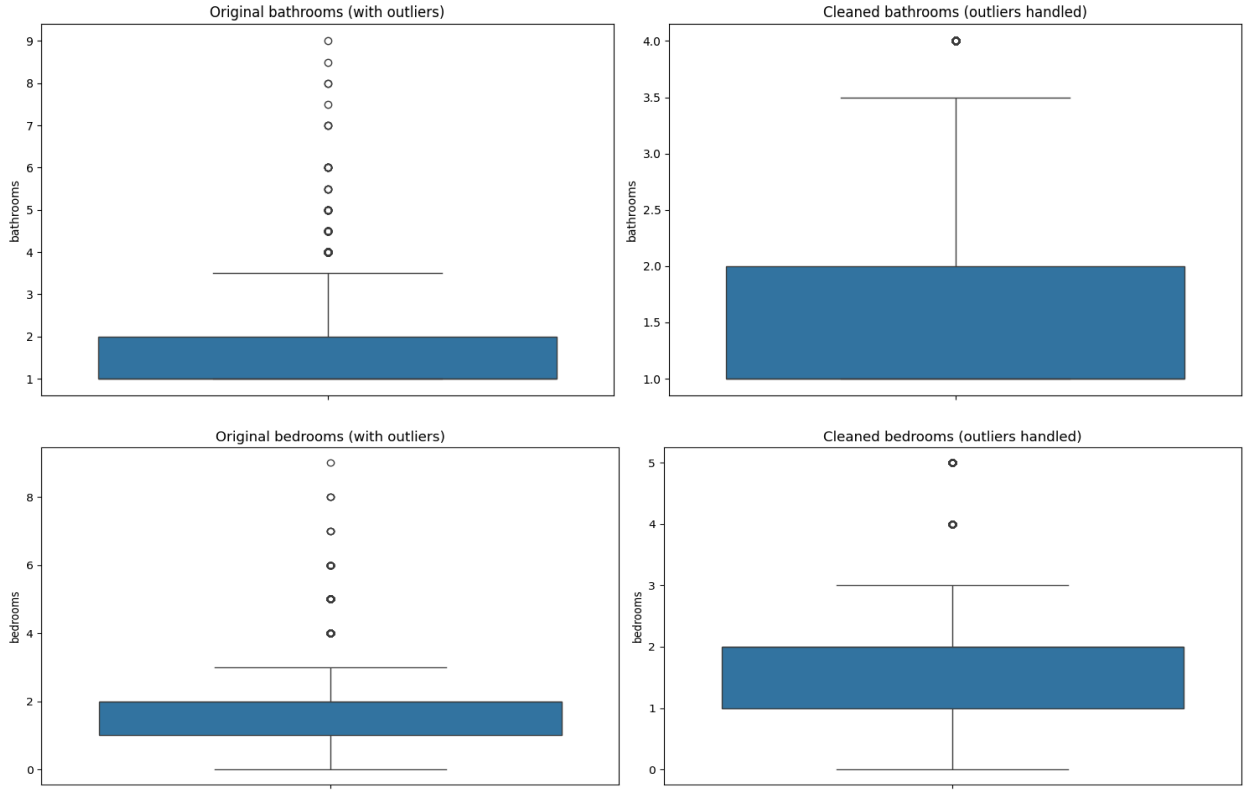
A summary table quantified the number and percentage of outliers detected for each relevant feature.

Dataset Columns	Number of Outliers	Outliers Range	Percentage of Outliers	IQR Bounds
bathrooms	202	4.00 to 9.00	0.20%	[-0.50, 3.50]
bedrooms	1832	4.00 to 9.00	1.80%	[-0.50, 3.50]

price	4622	2970.00 to 52500.00	4.66%	[-162.50, 2969.50]
square feet	2836	101.00 to 40000.00	2.86%	[150.00, 1694.00]
latitude	89	19.57 to 64.83	0.09%	[24.43, 49.27]
longitude	86	-159.37 to -147.72	0.09%	[-145.68, -36.72]

Outliers Handling:

- Outlier handling involved removing data points falling outside the calculated 1.5 * IQR range for selected features.
- This removal targeted the 'bathrooms', 'bedrooms', 'price', 'square_feet', 'latitude', and 'longitude' columns specifically.
- The process was applied *sequentially*: outliers were removed based on 'bathrooms', then the *resulting* data was filtered for 'bedrooms', and so on.
- This iterative filtering progressively removed rows containing extreme values according to each specified column's bounds.
- The final clean_data DataFrame resulted from this sequential removal across the key features



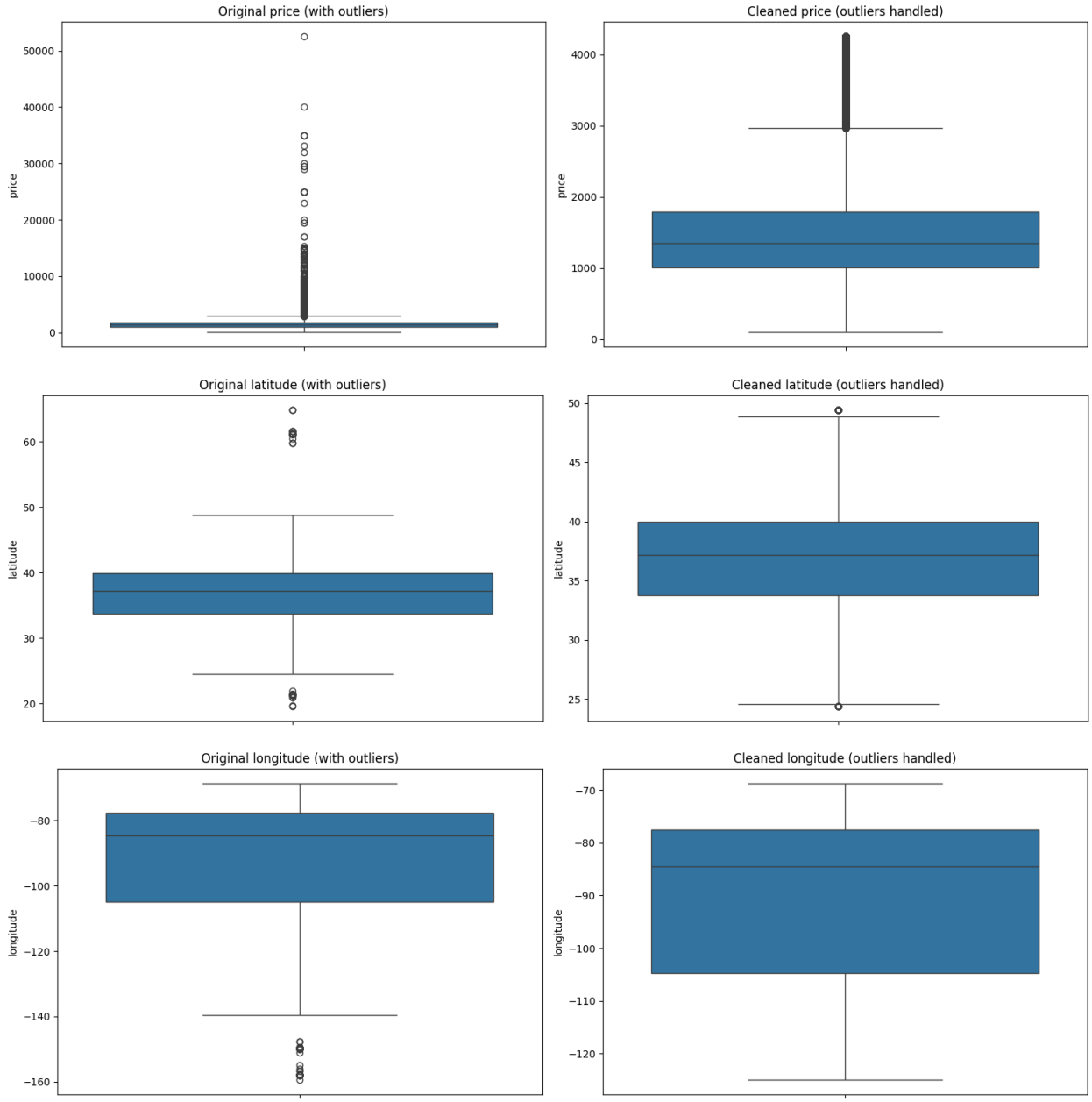


Figure 9: Box plot for features with outliers and outliers handled

Key Observation:

From Figure 9, we can see that almost all of the outliers has been handled properly. Here below i have explained the key observations:

1. Outlier reduction considerably reduced the range, particularly for 'price', by eliminating very high values from the original data.

2. The technique effectively deleted the most extreme places determined by the IQR method for 'latitude', 'longitude', and 'bedrooms'.

3. After cleaning, several points previously within the broader range now appear as outliers relative to the tighter distribution of the handled data (for example, in 'latitude', 'price', 'bedrooms').

METHODOLOGY / APPROACH:

Handling Data Leakage:

During early experimentation, the price_per_sqft and price_per_bedroom features were engineered by dividing the price by square feet and per bedroom. While this feature boosted model accuracy significantly (to over 95%), it was deemed leaky, as it directly encoded the label's information. This feature was removed from the final pipeline to preserve model fairness and avoid target leakage.

Feature Engineering:

Feature engineering was primarily focused on transforming the target variable (price) into categorical tiers. The continuous price variable was transformed into three distinct categories:

- Low: Prices below the 33rd percentile
- Medium: Prices between the 33rd and 66th percentiles
- High: Prices above the 66th percentile

A derived numerical feature, room_density, bed_bath_ratio, and sqft_per_bedroom
was introduced and defined as:
room_density = (bedrooms + bathrooms) / square_feet
bed_bath_ratio = bedrooms/bathrooms
Sqft_per_bedroom = square_feet / bedrooms

These metrics capture the spatial density and ratio of a listing, helping differentiate between compact and spacious layouts. Empirical analysis showed that this feature improved model performance by providing additional signal, especially in cases where absolute size metrics (square feet) failed to capture layout efficiency.

Feature Selection:

we selected the following features based on domain knowledge and initial data exploration:

Numeric Features: bedrooms, bathrooms, square_feet, bed_bath_ratio, sqft_per_bedroom latitude, longitude
Categorical Features: cityname, state, pets_allowed, price_type

Feature Scaling:
Numerical features were scaled using the StandardScaler to standardize their ranges. It also ensure all features contribute equally to the models
The formula applied is as follows:

$$x_{\text{scaled}} = \frac{x - \mu}{\sigma},$$

where μ is the mean and σ is the standard deviation of the feature.

Encoding Categorical Variables:
Categorical features were encoded into numerical values to make them suitable for machine learning models:
• Binary Features: Encoded using Label Encoding.
• Multi-Valued Features: Encoded using One-Hot Encoding with the first category dropped.

Model Selection Strategy:

We employed a combination of ensemble-based, linear, probabilistic, and instance-based models to address the classification task of predicting rental price tiers. This diverse selection enabled a holistic evaluation of model interpretability and predictive performance, especially given the high dimensionality introduced by one-hot encoding.

The **Random Forest Classifier** was selected as a primary model due to its robustness to multicollinearity and strong performance in high-dimensional, sparse datasets. As an ensemble method, it builds multiple decision trees and aggregates their outputs, providing stability and feature importance insights. We trained a 30-estimator forest with parallelization (`n_jobs = -1`) to achieve an efficient trade-off between training time and accuracy.

We implemented a **Decision Tree Classifier** with a constrained depth of 4 to serve as a baseline. While this model lacks complexity, it offers transparency and interpretability, allowing us to benchmark performance improvements gained from more advanced methods.

We also evaluated a **Logistic Regression** model to understand the effectiveness of a linear classifier. Its simplicity and interpretability made it useful for identifying linear relationships between engineered features and the target variable.

The **K-Nearest Neighbors (KNN)** model introduced a non-parametric perspective to our experimentation. KNN predicts based on similarity and is capable of capturing non-linear boundaries. However, its performance is sensitive to feature scaling and high-dimensionality, factors we accounted for during preprocessing.

To explore boosting techniques, we utilized **AdaBoost**, configuring it with decision stumps (shallow trees) as base learners. Hyperparameters such as `n_estimators` and `learning_rate` were tuned via GridSearchCV with stratified 5-fold validation. AdaBoost demonstrated strong class-level performance and highlighted the benefit of adaptive weighting in ensemble learning.

Finally, **Gaussian Naive Bayes (GNB)** was incorporated for its computational efficiency and suitability for continuous feature distributions. The smoothing parameter `var_smoothing` was optimized over a logarithmic scale, and preprocessing was tailored to ensure compatibility with GNB's assumptions.

All models were validated using **StratifiedKFold (k=5)** to ensure balanced class representation across folds. Performance was assessed using accuracy and macro-averaged F1 score to account for potential imbalances among the Low, Medium, and High classes.

IMPLEMENTATION AND ANALYSIS:

Model Training Procedure

We implemented six distinct models using standardized training workflows to evaluate the predictive performance of different classification algorithms on apartment rental pricing. All models were developed using Python and the scikit-learn library, with consistent training and testing splits to ensure comparability. Stratified 80/20 train-test splitting was applied to preserve class distributions across Low, Medium, and High price categories.

1. Random Forest Classifier

The Random Forest model was configured with 30 estimators, using parallel training (`n_jobs = -1`) for efficiency. This ensemble method was selected for its robustness to feature multicollinearity and its effectiveness on high-dimensional, sparse input produced by one-hot encoding.

```
Unset
rf_model = RandomForestClassifier(n_estimators=30, random_state=42, n_jobs=-1)
```

2. Decision Tree Classifier

A shallow Decision Tree (max depth = 4) was implemented as a lightweight benchmark model. This baseline allows us to observe the gains from ensembling methods while maintaining interpretability.

```
Unset
dt_model = DecisionTreeClassifier(max_depth=4, random_state=42)
```

3. K-Nearest Neighbors (KNN)

KNN was trained using a pipeline consisting of median imputation, standard scaling, PCA reduction (15 components), and SMOTE to address class imbalance. A grid search over `n_neighbors`, `metric`, and `weights` was performed using 5-fold cross-validation.

```
Unset
param_grid = {
    'n_neighbors': [5, 7, 9, 11],
    'weights': ['uniform', 'distance'],
    'metric': ['euclidean', 'manhattan']
}
```

4. Logistic Regression

This linear model was trained using polynomial interaction terms and SMOTE. Preprocessing included median imputation and standard scaling. Cross-entropy loss was minimized using the LBFGS solver with `max_iter = 1000`.

```
Unset
classifier = LogisticRegression(max_iter=1000, solver='lbfgs', random_state=42)
```

5. Naive Bayes (GaussianNB)

Gaussian Naive Bayes was selected for its efficiency and simplicity, especially suited to continuous numerical features. Numerical attributes were standardized, and categorical features were one-hot encoded. Hyperparameter tuning focused on `var_smoothing`, optimized via grid search over log-scaled values.

```
Unset
param_grid = {'var_smoothing': np.logspace(-9, -1, 9)}
```

6. AdaBoost Classifier

AdaBoost was built using shallow Decision Trees (`max_depth = 1`) as base estimators. Grid search over `n_estimators` and `learning_rate` identified optimal configurations. Final models were trained with the best hyperparameters and evaluated on accuracy and confusion matrix performance.

```
Unset
base_estimator = DecisionTreeClassifier(max_depth=1)

AdaBoostClassifier(...)
```

Across all models, stratified cross-validation (k=5) was used for fair and consistent model comparison. Performance evaluation metrics include accuracy, F1-score per class, confusion matrices, and ROC-AUC curves where applicable. This multi-model setup enables a robust comparative framework to assess model fit, generalization, and suitability for rental price tier classification.

RESULTS AND EVALUATION:

Evaluation Settings

To ensure fair comparison across models, we adopted a consistent evaluation framework. All classifiers were assessed using **5-fold stratified cross-validation**, preserving class distribution during training and validation splits. This approach provides a more reliable estimate of model generalization, especially in class imbalance.

For performance measurement, we relied on a combination of standard classification metrics:

- **Accuracy** — overall correctness of predictions
- **Macro F1-score** — average F1 across all classes, giving equal weight to each category
- **Class-wise F1-scores** — to reveal model sensitivity and precision for each rental tier
- **Confusion matrices** — visualizing misclassification patterns
- **ROC-AUC curves** — where applicable, to evaluate probabilistic confidence across classes

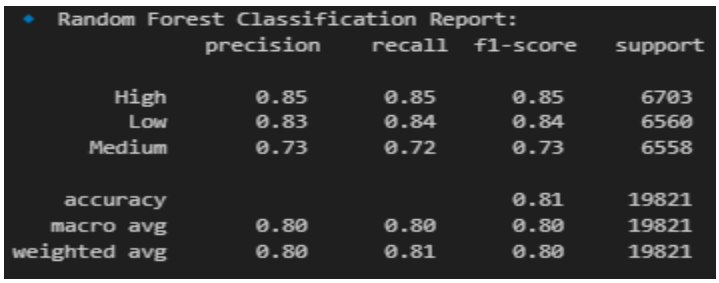
To mitigate class imbalance, **SMOTE (Synthetic Minority Over-sampling Technique)** was applied during model training in selected pipelines (e.g., KNN, Logistic Regression). This ensured each model received a balanced representation of price categories during learning.

Preprocessing pipelines included standard scaling for numerical inputs, one-hot encoding for categorical variables, and dimensionality reduction (via PCA) where appropriate. All preprocessing was encapsulated within scikit-learn Pipelines to avoid data leakage.

Model Evaluation and Performance Analysis:

This section presents a detailed evaluation of the six supervised learning models applied to the rental price tier classification task. The models were trained on a cleaned and feature-engineered dataset of 99,105 apartment listings, using a stratified 80/20 train-test split to preserve class distribution. Evaluation metrics include accuracy, class-wise F1 scores, confusion matrices, and ROC-AUC analysis..

The Random Forest model demonstrated strong generalization with a cross-validation accuracy of 80.38% ± 0.44% and a test accuracy of 81%. The confusion matrix indicated misclassifications were concentrated in the Medium class, often confused with its adjacent categories. Feature importance analysis revealed that square footage, room density, and geographic indicators were the most influential.



Random Forest Classification Report:

	precision	recall	f1-score	support
High	0.85	0.85	0.85	6703
Low	0.83	0.84	0.84	6560
Medium	0.73	0.72	0.73	6558
accuracy			0.81	19821
macro avg	0.80	0.80	0.80	19821
weighted avg	0.80	0.81	0.80	19821

Decision Tree Classifier. The shallow Decision Tree (max depth = 4) served as a baseline model. It yielded a significantly lower performance, with a test accuracy of 54% and macro F1 of 0.53. The confusion matrix illustrated over-prediction of the Low class and substantial confusion in the Medium category. This performance gap highlighted the limitations of simple models and the value of ensemble techniques.

Decision Tree Classification Report:

	precision	recall	f1-score	support
High	0.77	0.45	0.57	6703
Low	0.52	0.83	0.64	6560
Medium	0.42	0.34	0.38	6558
accuracy			0.54	19821
macro avg	0.57	0.54	0.53	19821
weighted avg	0.57	0.54	0.53	19821

K-Nearest Neighbors(KNN) was trained with SMOTE-balanced training data and tuned via GridSearchCV. Optimal parameters (k=9, metric='manhattan', weights='distance') achieved a test accuracy of 84%. The model's performance improved significantly with scaling and dimensionality reduction via PCA. ROC curves and heatmaps demonstrated strong class separation and stability.

2025-04-23 22:02:36,576 - INFO - Classification Report:

	precision	recall	f1-score	support
High	0.84	0.85	0.84	6704
Low	0.84	0.83	0.84	6561
Medium	0.73	0.73	0.73	6556
accuracy			0.80	19821
macro avg	0.80	0.80	0.80	19821
weighted avg	0.80	0.80	0.80	19821

Logistic Regression incorporated interaction features using PolynomialFeatures (degree=2) and SMOTE. The model achieved test accuracy of 73.76% with well-balanced F1 scores across all categories. Feature importance plots based on learned coefficients offered interpretability, revealing strong contributions from square footage and city/state indicators. ROC-AUC plots confirmed the model's ability to separate price categories effectively.

2025-04-22 20:59:58,206 - INFO - Classification Report:

	precision	recall	f1-score	support
High	0.82	0.79	0.80	6821
Low	0.77	0.78	0.78	6573
Medium	0.63	0.63	0.63	6499
accuracy			0.74	19893
macro avg	0.74	0.74	0.74	19893
weighted avg	0.74	0.74	0.74	19893

Naive Bayes(GaussianNB) achieved a test accuracy of 61%, the highest among all models. Hyperparameter tuning (var_smoothing) was conducted using 5-fold Stratified Cross-Validation. Despite its strong performance, the model assumes feature independence, which may oversimplify complex feature interactions. Confusion matrix and class-wise F1 scores remained balanced, with minimal overfitting.

Classification Report:

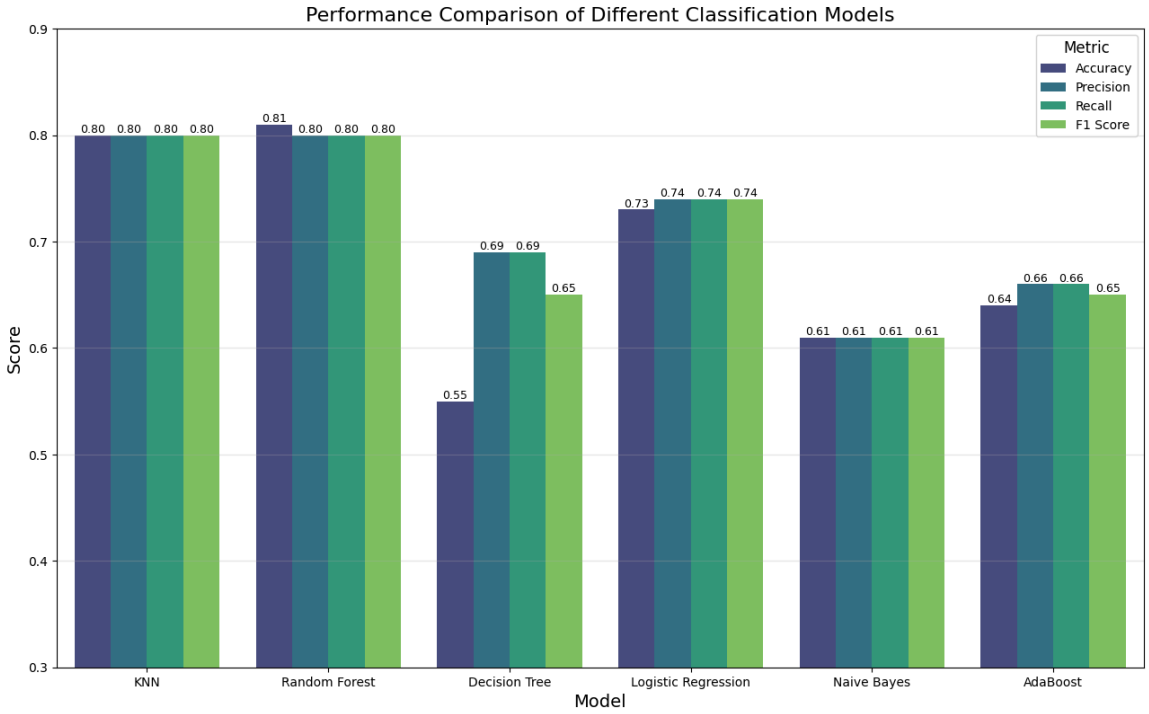
	precision	recall	f1-score	support
High	0.65	0.67	0.66	6540
Low	0.66	0.68	0.67	6561
Medium	0.51	0.49	0.50	6720
accuracy			0.61	19821
macro avg	0.61	0.61	0.61	19821
weighted avg	0.61	0.61	0.61	19821

AdaBoost with a base estimator of depth-1 Decision Trees (stumps) reached a test accuracy of 65%. Grid search over n_estimators and learning_rate identified optimal values (100, 0.5). ROC-AUC curves and confusion matrix confirmed the model's stability across classes. Feature importance analysis indicated meaningful usage of both structural and categorical features.

Classification Report:

	precision	recall	f1-score	support
High	0.73	0.70	0.72	6540
Low	0.71	0.67	0.69	6561
Medium	0.52	0.57	0.55	6720
accuracy			0.65	19821
macro avg	0.66	0.65	0.65	19821
weighted avg	0.66	0.65	0.65	19821

The results demonstrate the importance of preprocessing (e.g., SMOTE, PCA, scaling) and model selection. While Naive Bayes delivered competitive accuracy, ensemble methods like AdaBoost and Random Forest offered robust, interpretable alternatives with strong performance. The Decision Tree baseline underscored the importance of model complexity and hyperparameter tuning in structured tabular data scenarios.



	Accuracy	Precision	Recall	F1 Score
KNN	0.8	0.8	0.8	0.8
Random Forest	0.81	0.8	0.8	0.8
Decision Tree	0.55	0.69	0.69	0.65
Logistic Regression	0.73	0.74	0.74	0.74
Naive Bayes	0.61	0.61	0.61	0.61
AdaBoost	0.64	0.66	0.66	0.65

Figure 10: Performance Comparison of Different Classification Models

Performance Summary:

Model	**Feature Importance**	**Test Accuracy**	**Cross-Validation Accuracy**	**Macro F1 Score**
Random Forest	- square_feet - room_density - bathrooms	81%	80.38%	0.80
KNN	-	80%	98%	0.80
Logistic Regression	- cityname - state	73.76%	75.53%	74%

AdaBoost	- longitude - latitude - state_CO - square_feet	65%	64.94%	0.65
Naive Bayes	- square_feet - bathrooms - bedrooms - latitude	61%	61.28%	0.61
Decision Tree	- square_feet - cityname - state	54%	54.49%	0.53

Best Model:

Random Forest significantly outperformed the Decision Tree, KNN, Logistic Regression, AdaBoost, and Naive Bayes in both accuracy and F1 score. The ensemble structure of Random Forest provided better generalization and less overfitting on sparse One-Hot encoded features.

Class-wise F1 Scores (Random Forest)

Class	**Precision**	**Recall**	**F1 Score**
High	0.85	0.85	0.85
Low	0.83	0.84	0.84
Medium	0.73	0.72	0.73

The model performs best on the “High” and “Low” tiers, with slightly lower performance on the “Medium” category due to price overlap near quantile boundaries.

Confusion Matrix of Best Model:

This confusion matrix for the Random Forest classifier implies strong performance in determining the various categories ("High," "Low," and "Medium") of rental price. Most predictions fall along the diagonal, indicating correct classifications: 5,708 "High," 5,519 "Low," and 4,730 "Medium" listings were accurately identified. Misclassifications are most common in the "Medium" category, which is often confused with "High" (867 cases) and "Low" (961 cases),

while "High" and "Low" are less frequently mistaken for each other. The model is effective, but separating "Medium" from the other categories is challenging.

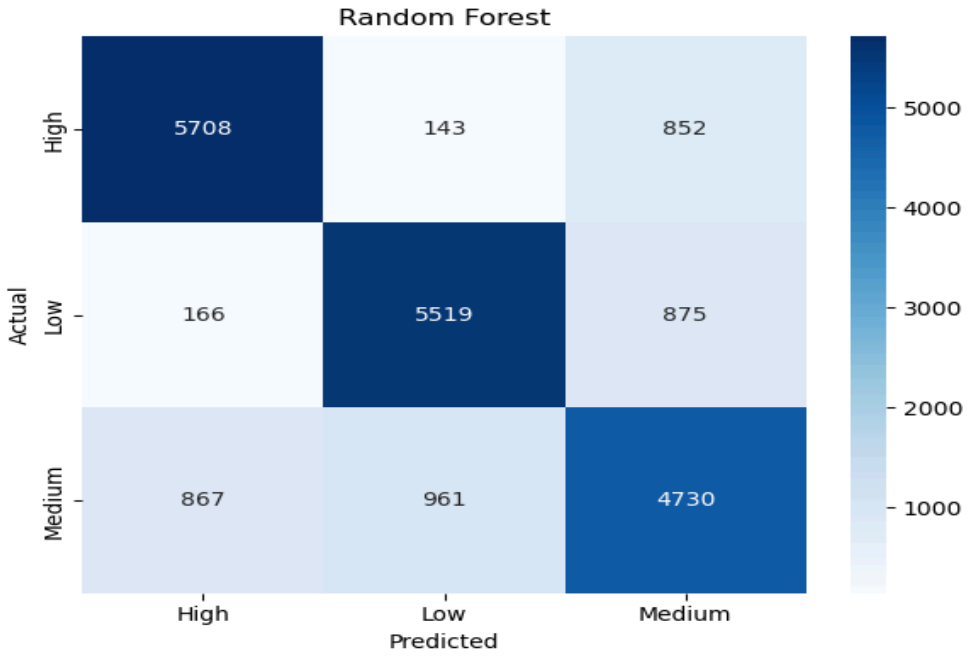


Figure 11: Confusion Matrix of Random Forest

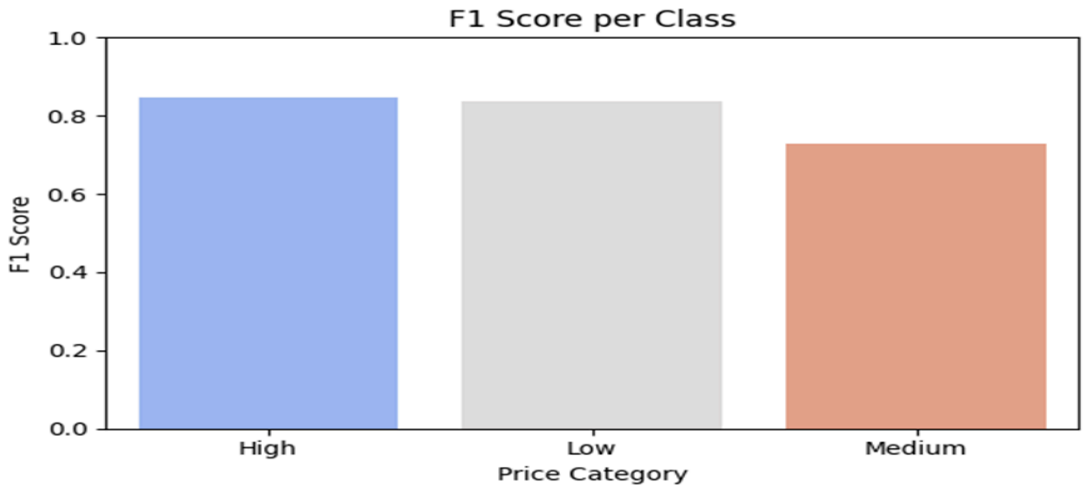


Figure 12: F1 Score per Class of Random Forest

Observation:

The F1 scores for the "High" and "Low" price categories are both strong and nearly identical, indicating balanced model performance for these classes. The "Medium" category has a noticeably lower F1 score, suggesting it is more challenging for the model to predict accurately.

CONCLUSION:

While the Random Forest model demonstrated strong overall accuracy (81%) and balanced class-wise performance, there remain several limitations that point to future opportunities for improvement:

Regional Bias in Price Tiering:

The current approach uses global quantile cuts to define "Low," "Medium," and "High" price categories. This method fails to capture local market conditions. for example, a modest apartment in Manhattan may still be labeled "High." At the same time, a high-end listing in a rural state may appear as only "Medium." This disconnect limits the contextual fairness of the model.

Limited Feature Scope:

Our model focuses primarily on structural features such as square footage, bedroom and bathroom counts, and room density. However, key pricing drivers are absent, including:
renovation status and building age; floor level, lighting, and view quality; and distance to transportation, schools, or commercial hubs. Without these factors, the model lacks the granularity to distinguish between units with similar layouts but vastly different rental value.

Lack of Temporal Awareness:

Despite having access to listing timestamps, the model does not incorporate time-based features. Ignoring temporal dynamics such as seasonal price shifts or post-COVID market changes reduces the model's adaptability and long-term relevance.

FUTURE WORKS:

To address the above limitations and enhance both accuracy and fairness, future iterations should consider:

- ➔ Defining price tiers relative to local markets (city- or state-specific quantile cuts)
- ➔ Engineering richer contextual features (e.g., amenities_count, proximity to transit or schools)
- ➔ Incorporating time-based variables for temporal modeling
- ➔ Using SHAP values for scalable interpretability and fairness auditing
- ➔ Exploring ensemble architectures or multi-task learning for capturing regional pricing nuances

REFLECTION ON LEARNING:

Through this coursework, we've gained practical experience and deeper understanding in several areas:

Data Handling: Working with a large, imbalanced dataset taught us how to clean, transform, and engineer features that help models perform better.

Machine Learning Techniques: We got hands-on with multiple models like Decision Trees, KNN, Naive Bayes, and ensemble methods like Random Forest and AdaBoost. Comparing them gave us a clearer view of their strengths and trade-offs.

Evaluation Metrics: Beyond accuracy, we learned how metrics like precision, recall, and F1-score give a more complete picture—especially in imbalanced classification problems.

Visualization: Using matplotlib and seaborn, we visualized trends, relationships, and model results. These visual tools were essential for debugging and communication.

Problem-Solving: Turning raw rental data into actionable insights took collaboration, iteration, and strategic thinking—from cleaning to modeling to evaluation.

Tools & Libraries: Our proficiency with Python, pandas, scikit-learn, and visualization tools grew significantly. These skills are foundational for real-world data work.

What we enjoyed most was the **hands-on challenge of solving a real-world problem**—turning messy data into insight and model performance we could evaluate and improve.

References:

Data Source : https://archive.ics.uci.edu/dataset/555/apartment+for+rent+classified
Apartment for Rent Classified [Dataset]. (2019). UCI Machine Learning Repository. https://doi.org/10.24432/C5X623.