USA-REAL_ESTATE Price Prediction Project Documentation

LOADIND & SAMPLING DATA

- The dataset was loaded using pandas for analysis. The original dataset is large-scale; therefore, sampling was used to enable efficient iteration without compromising statistical validity.
- Initially, a sample size of 50K was used. To assess representativeness, different sample sizes were tested. Increasing the sample size did not materially change the observed patterns, indicating that the sampled data adequately represents the underlying distribution.

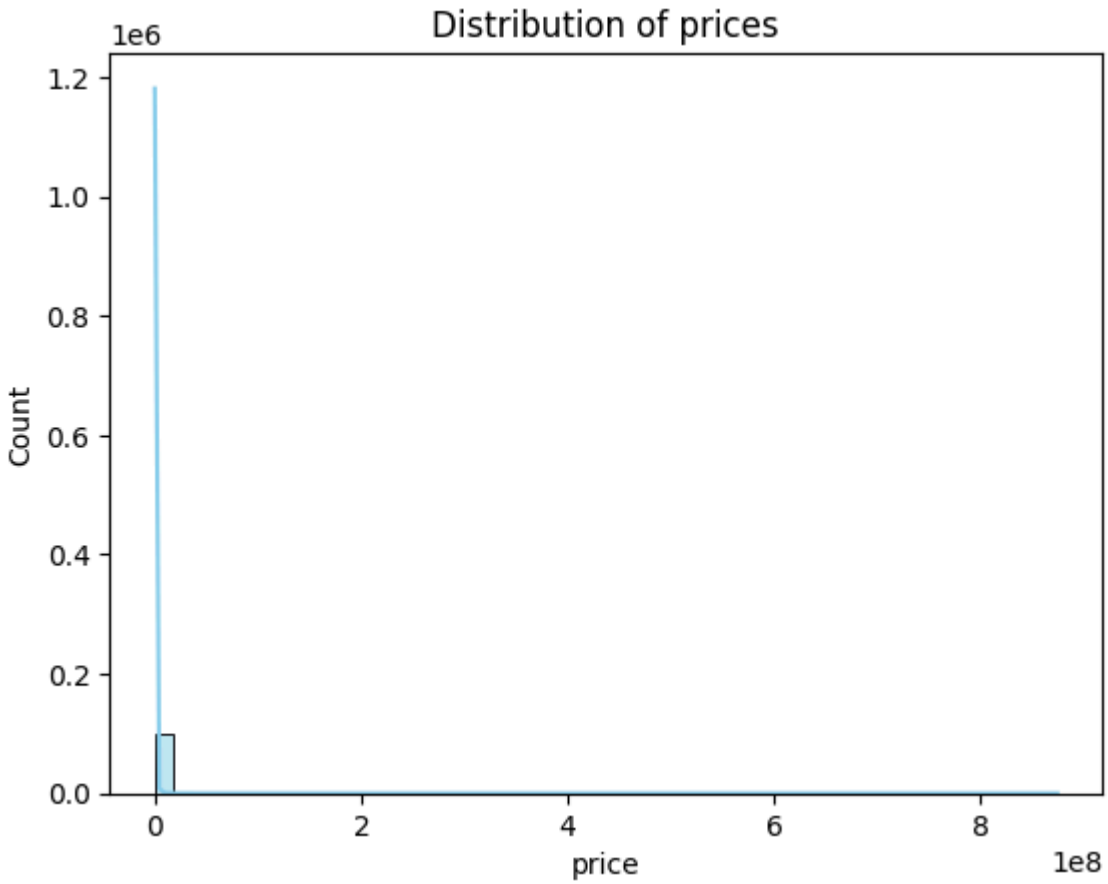
UNDERSTANDING DATA

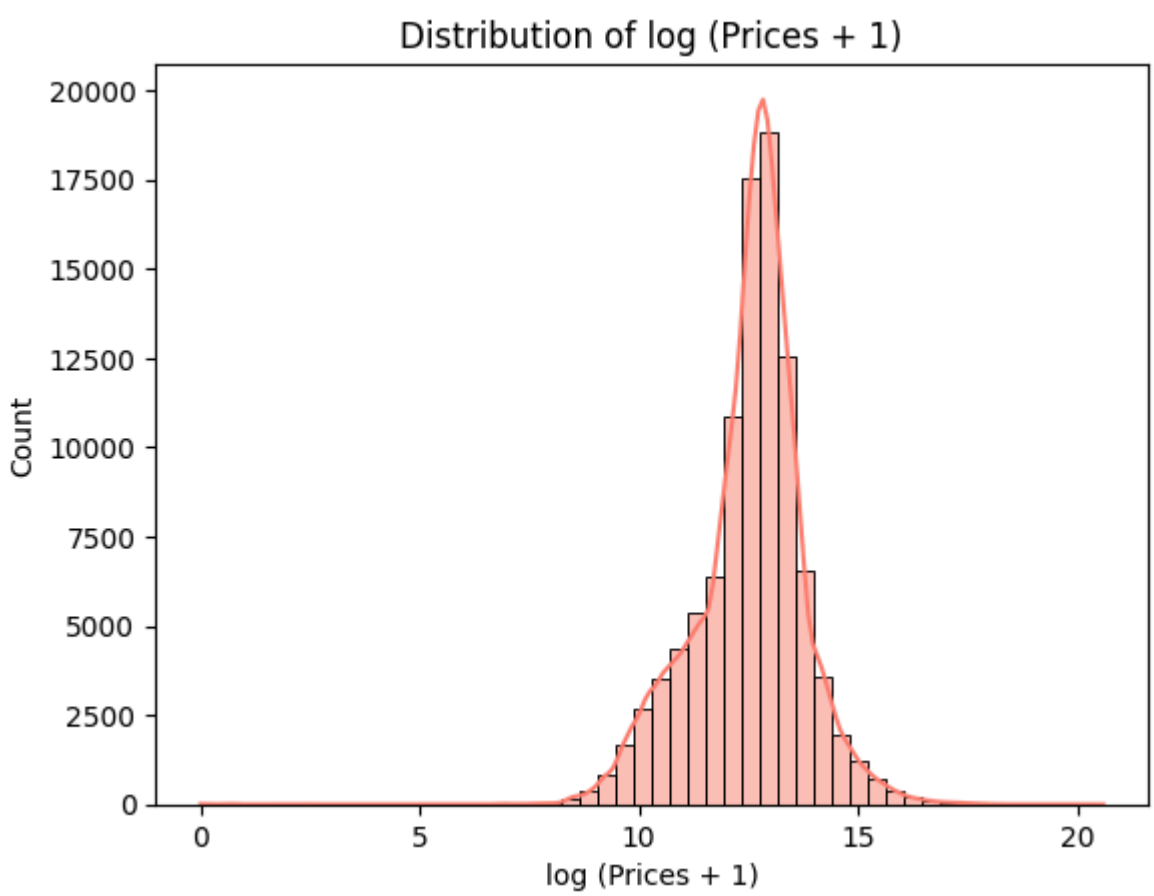
DATA UNDERSTANDING SUMMARY

In this step, the dataset was examined to understand its structure, feature types, summary statistics, and overall composition. Missing values and their percentages were analyzed across all columns, revealing that some features contain substantial missing data. This assessment helped identify potential data quality issues and informed decisions for exploratory data analysis and subsequent preprocessing steps.

Exploratory Data Analysis (EDA)

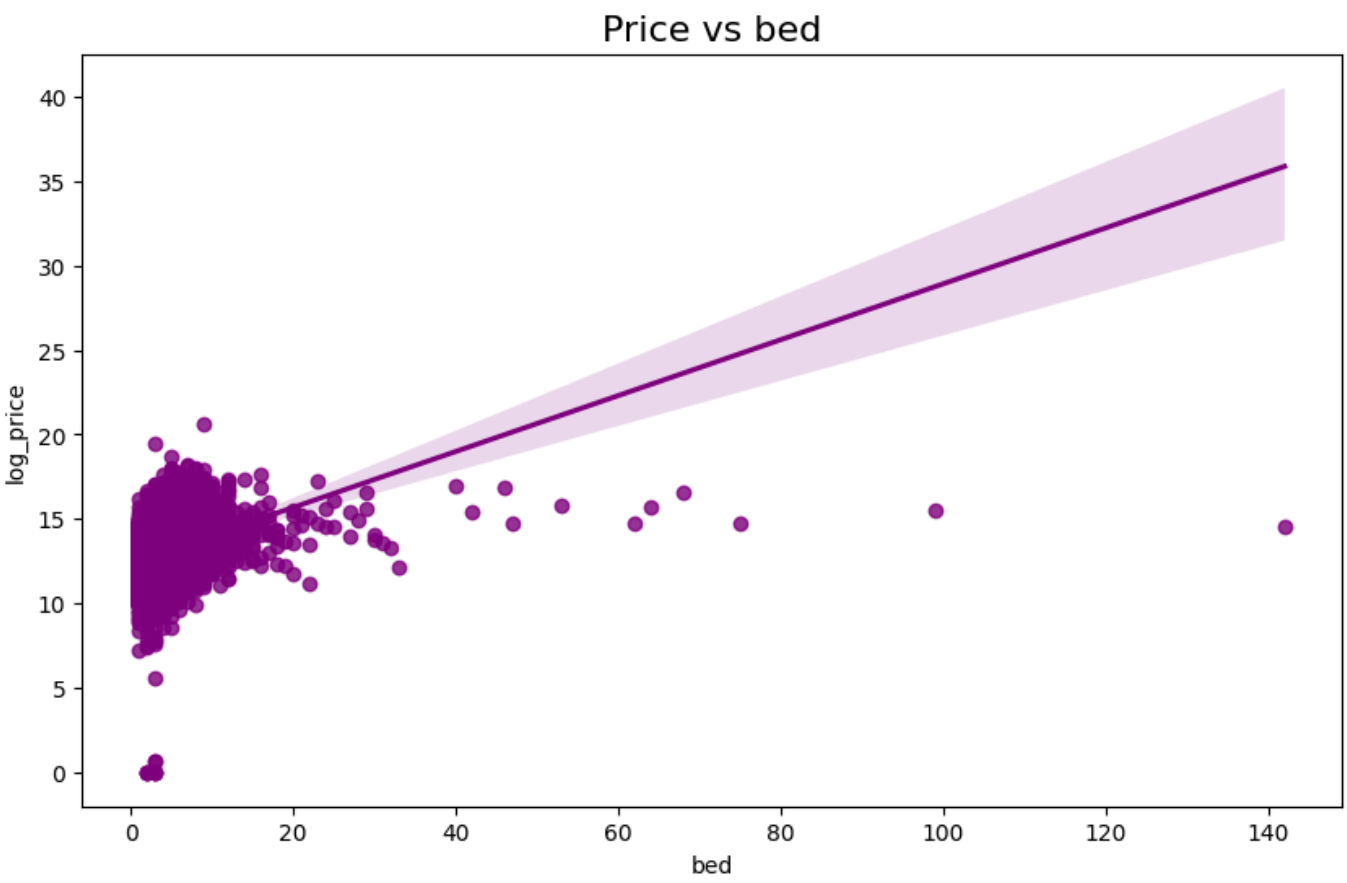
Exploratory data analysis was conducted to understand the distribution of the target variable, examine relationships between features and price, and identify patterns, trends, and outliers that may influence modeling decisions.

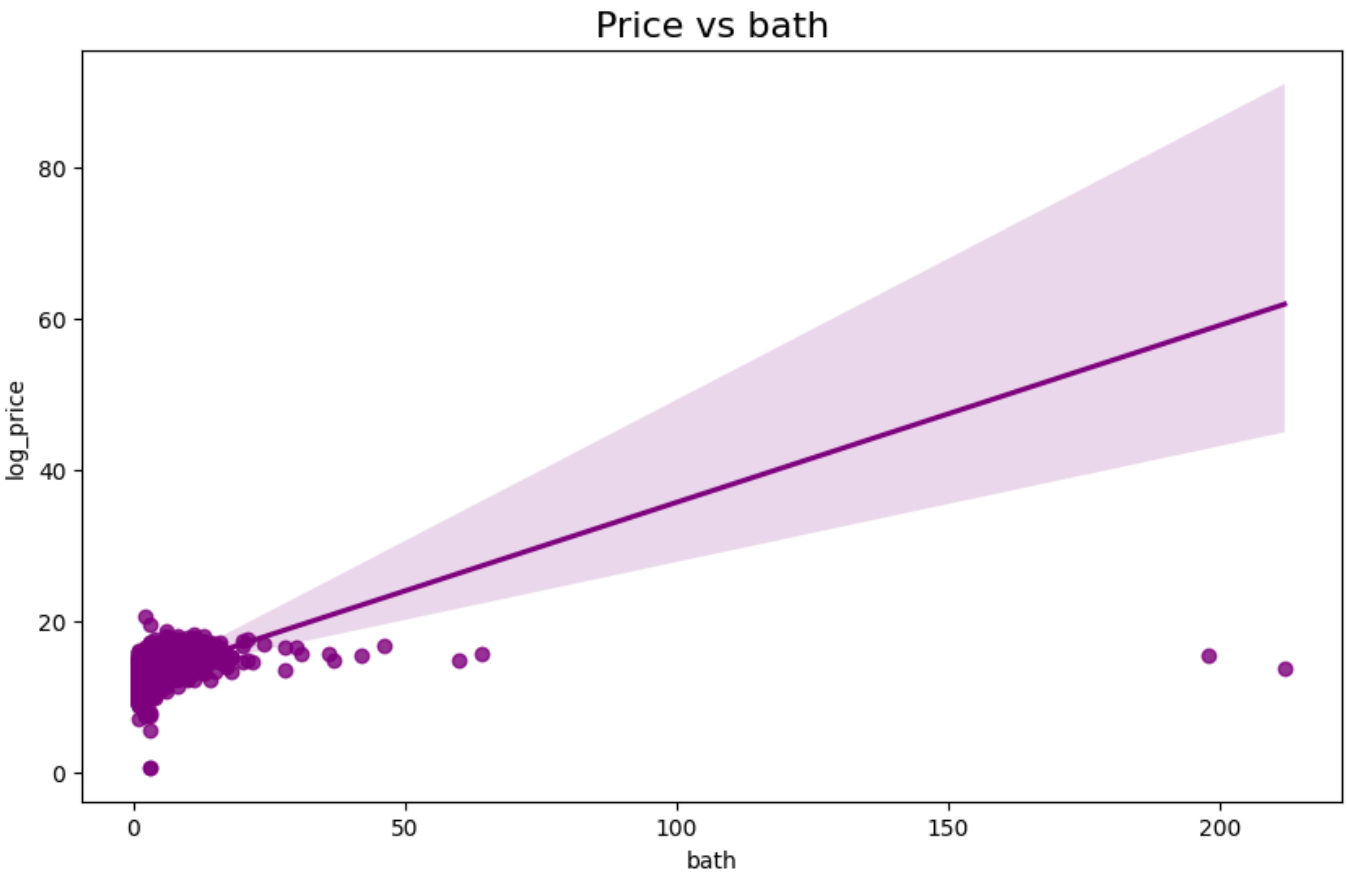


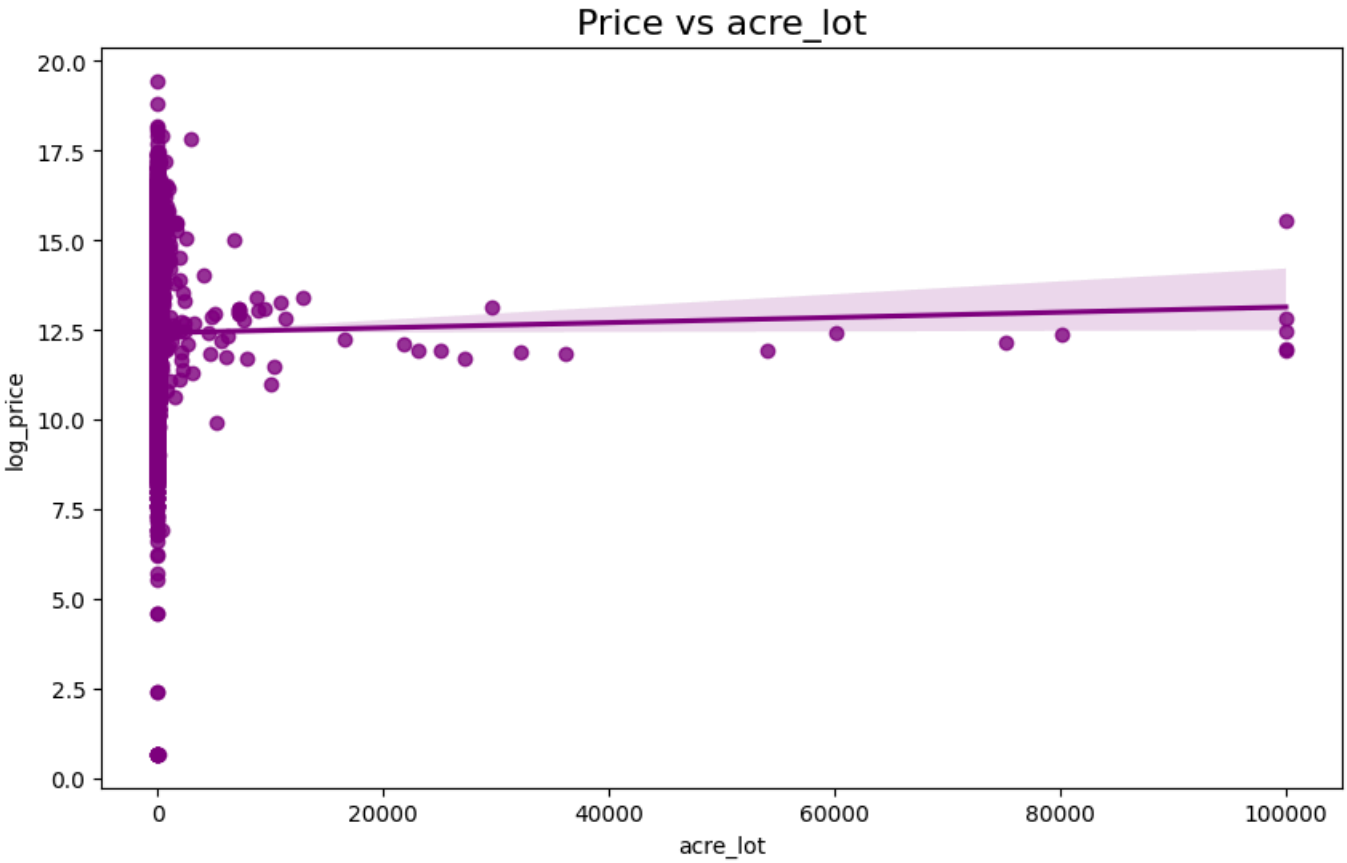


Finding:

The original price distribution was heavily right-skewed. Applying a log transformation reduced skewness and produced a more symmetric distribution, with most observations concentrated between 10 and 15 on the log scale.

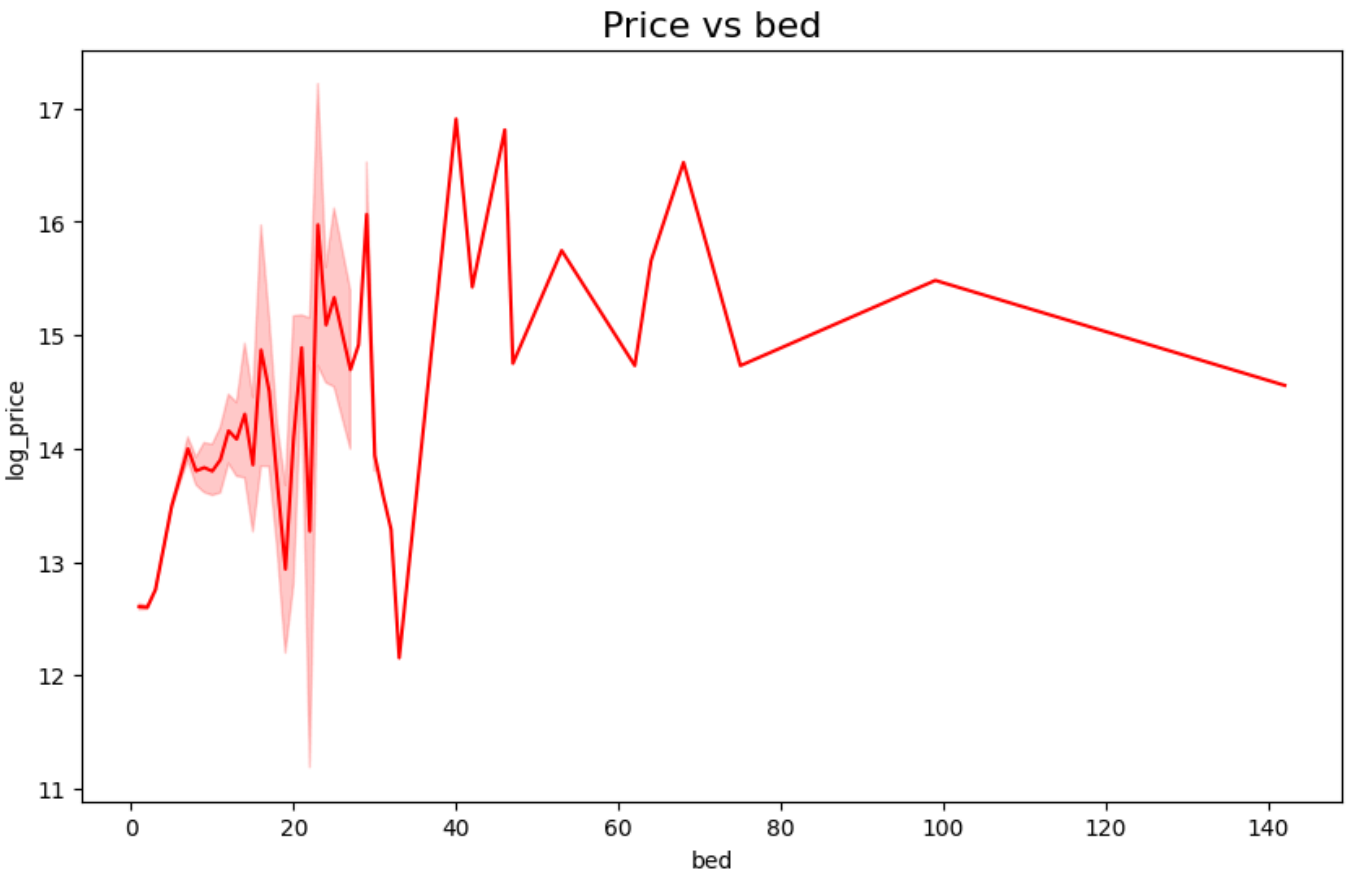

Price vs bed
log_price
bed
0
5
10
15
20
25
30
35
40
0
20
40
60
80
100
120
140

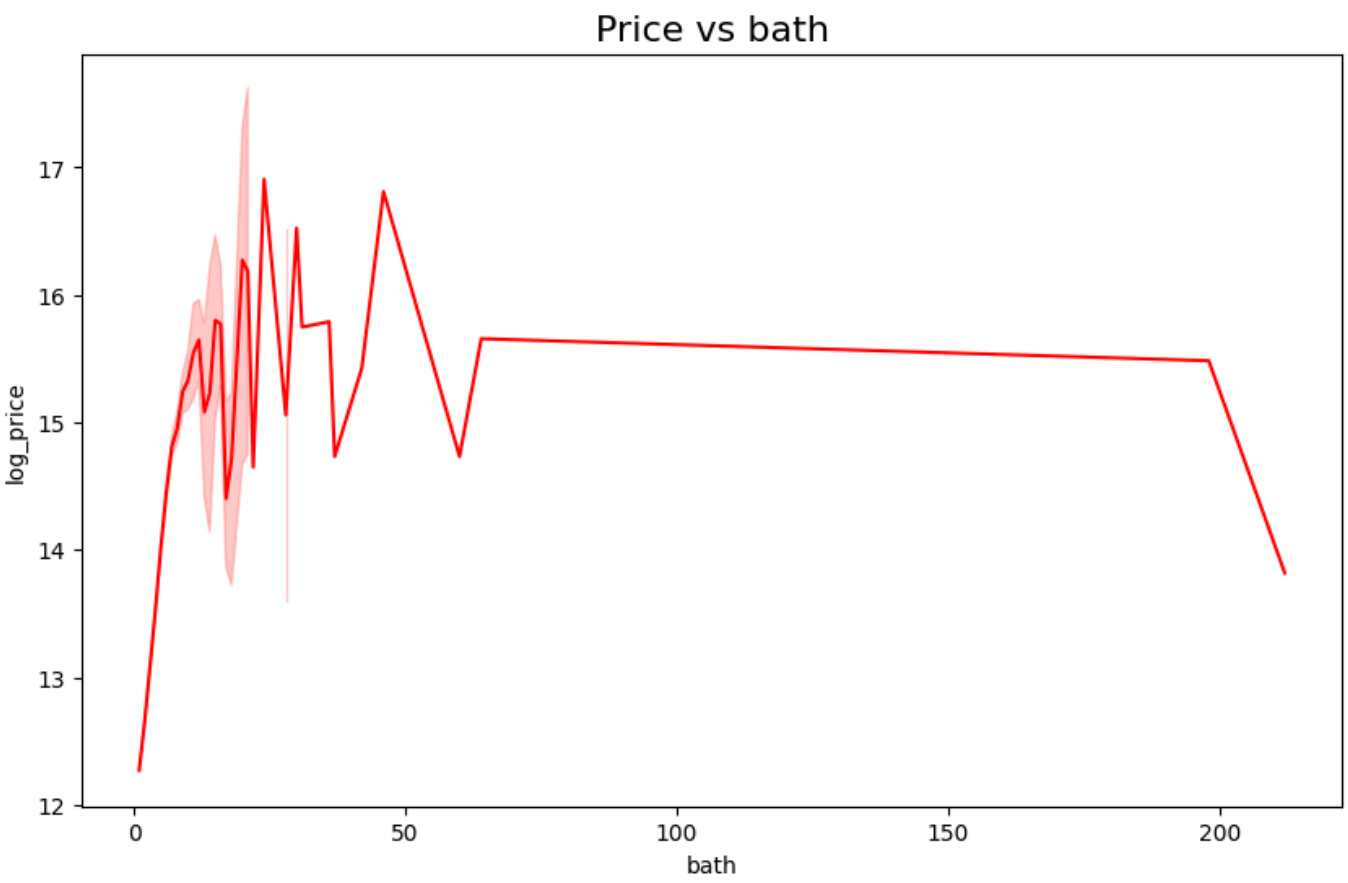

Price vs bath
log_price
bath
0
20
40
60
80
0
50
100
150
200


Price vs acre_lot
log_price
acre_lot
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
20000
40000
60000
80000
100000


Price vs brokered_by
log_price
brokered_by
20
15
10
5
0
0
20000
40000
60000
80000
100000


Price vs house_size
log_price
house_size
18
16
14
12
10
8
0
20000
40000
60000
80000
100000
120000


Price vs bed
log_price
bed
17
16
15
14
13
12
11
0
20
40
60
80
100
120
140


Price vs bath
log_price
bath
17
16
15
14
13
12
0
50
100
150
200

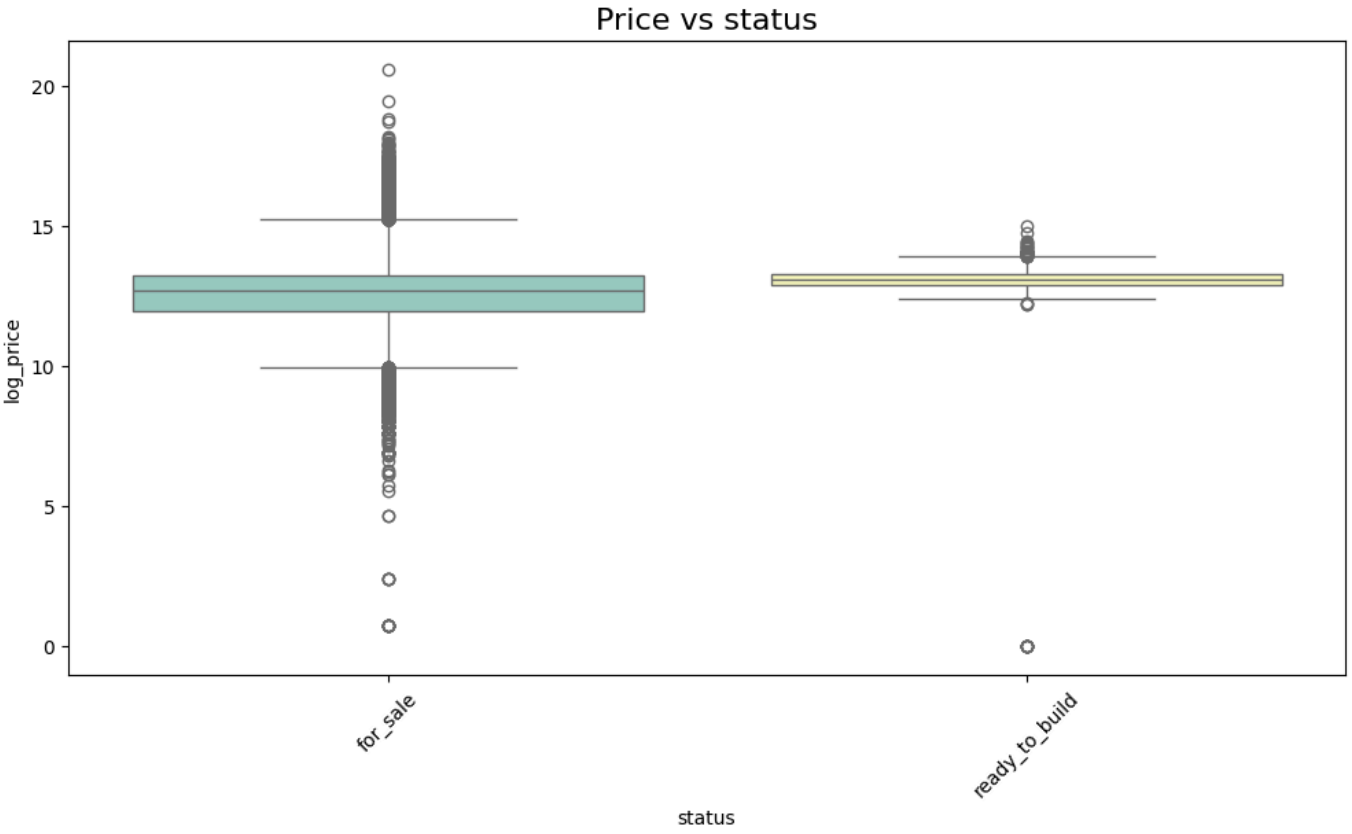

Price vs acre_lot
log_price
acre_lot
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
20000
40000
60000
80000
100000

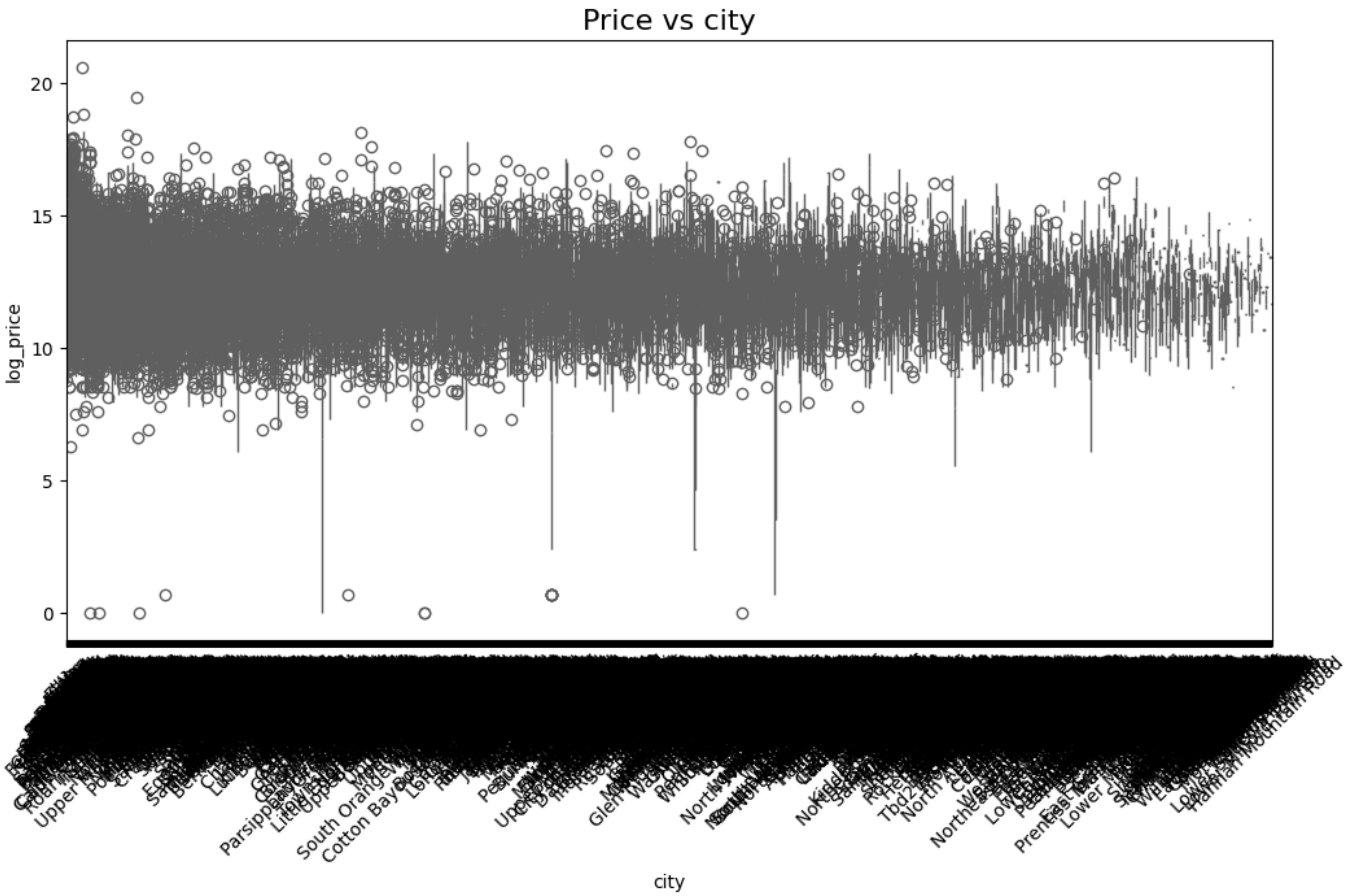
Finding:

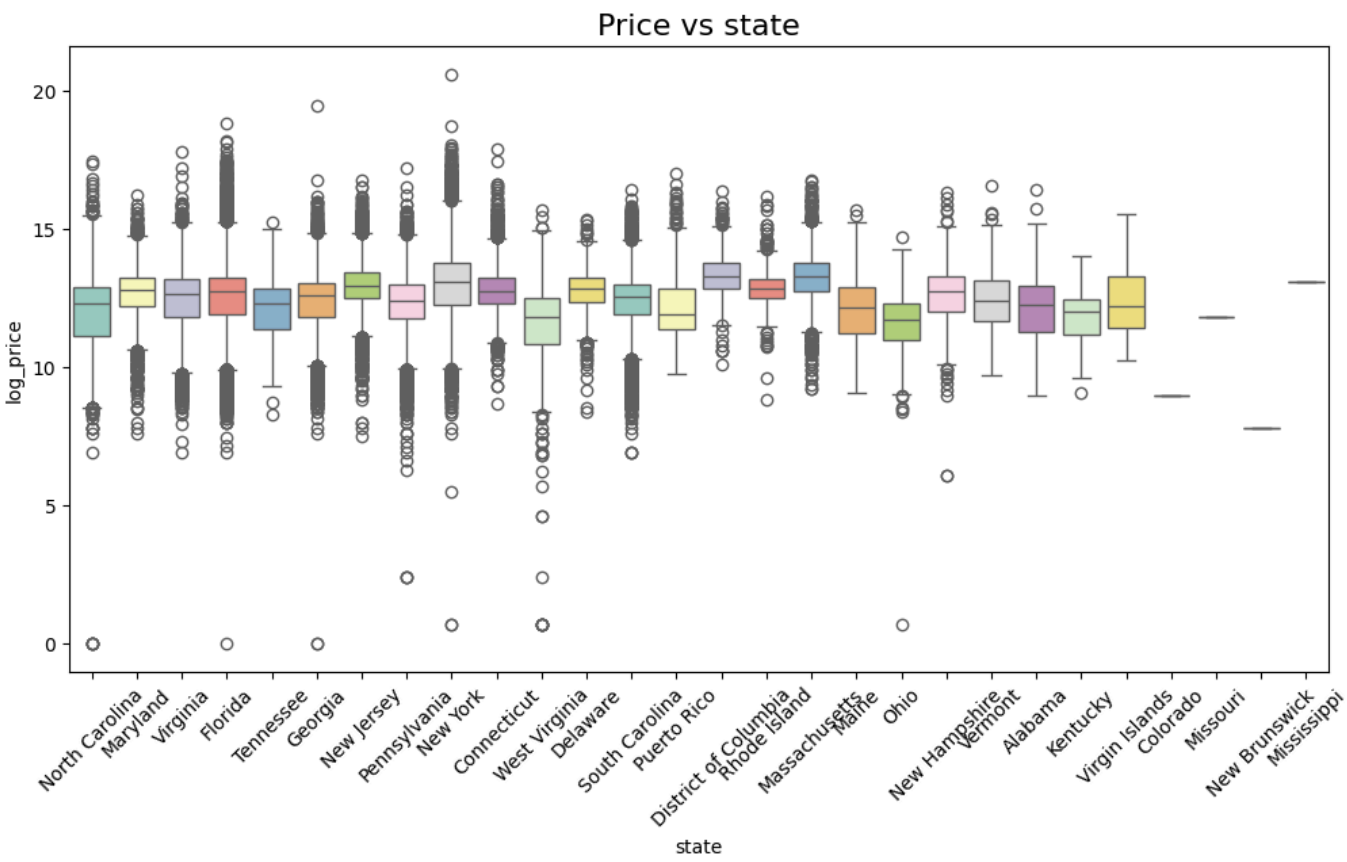
- house size vs log price | Analysis: House size shows a clear upward trend with log price
- bed vs log price | Analysis: Bedrooms show a mild positive effect on price
- bath vs log price | Analysis: Bathrooms show a stronger influence on price
- acre lot vs log price | Analysis: Acre lot shows minimal impact within observed range
- brokered_by vs log price | Analysis: Brokered_by shows no meaningful linear relationship with price

General Observation

Most numeric features are concentrated in lower ranges with a few outliers. House size and number of bathrooms show a clear upward trend with log price, bedrooms show a mild effect, while acre lot and brokered_by show minimal influence.


Price vs status
log_price
20
15
10
5
0
for_sale
ready_to_build
status


Price vs city
log_price
20
15
10
5
0
city



Finding:

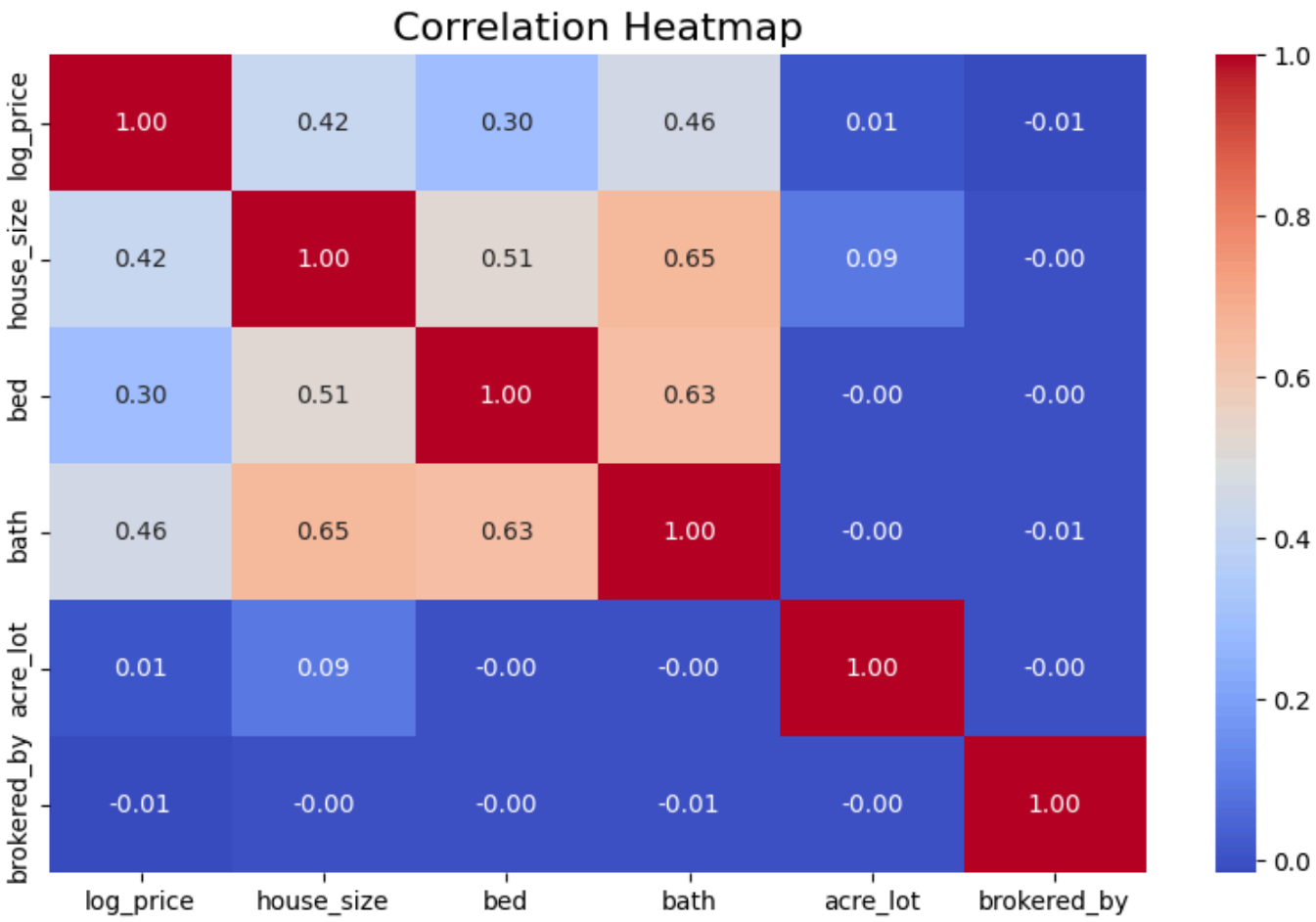
- Staus vs log price: The box plot and violin plot show that “For Sale” properties have a median log price between 10 and 15 with a few high and low outliers, while “Ready to Build” properties have a slightly higher median around 13 to 15 with minimal outliers. This indicates that the property status has a modest impact on price.
- City vs log price:
  The box plot of price across cities is not clearly visible due to the very high number of unique cities. However, the range of log prices across cities is approximately 9 to 17, suggesting that city-level variation in price is relatively small within this dataset.
- state vs log price: The box plot of price by state shows median log prices in a similar range (roughly 8.5–15.5) across states such as North Carolina, Maryland, and Virginia, indicating that state-level differences exist but are not extremely large in this sample.

Violin Plots Note:

Violin plots for city and state could not be rendered due to high cardinality, but the price distributions inferred from the box plots are consistent with the expected spread observed in the dataset.

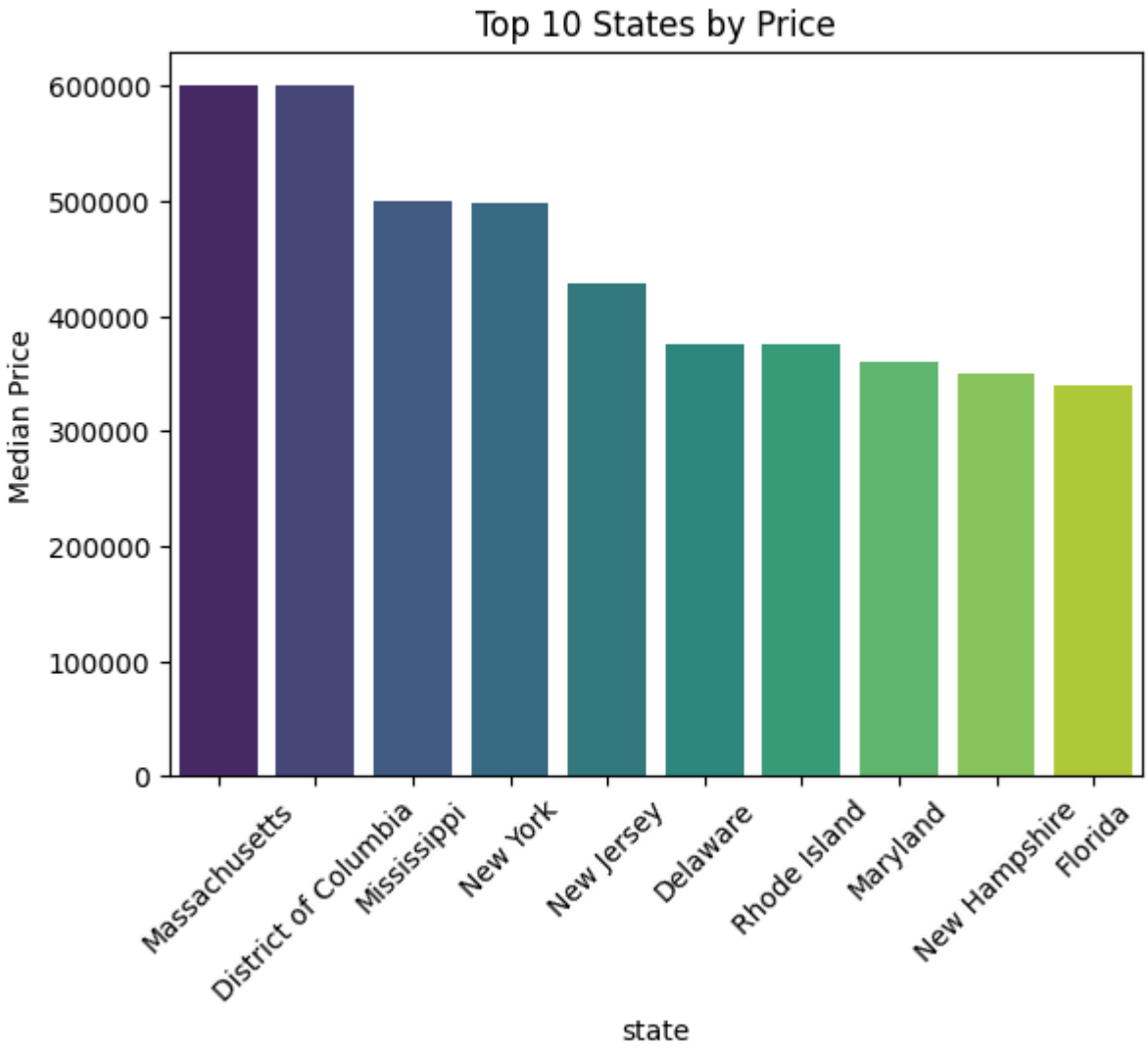
GENERAL OBSERVATION

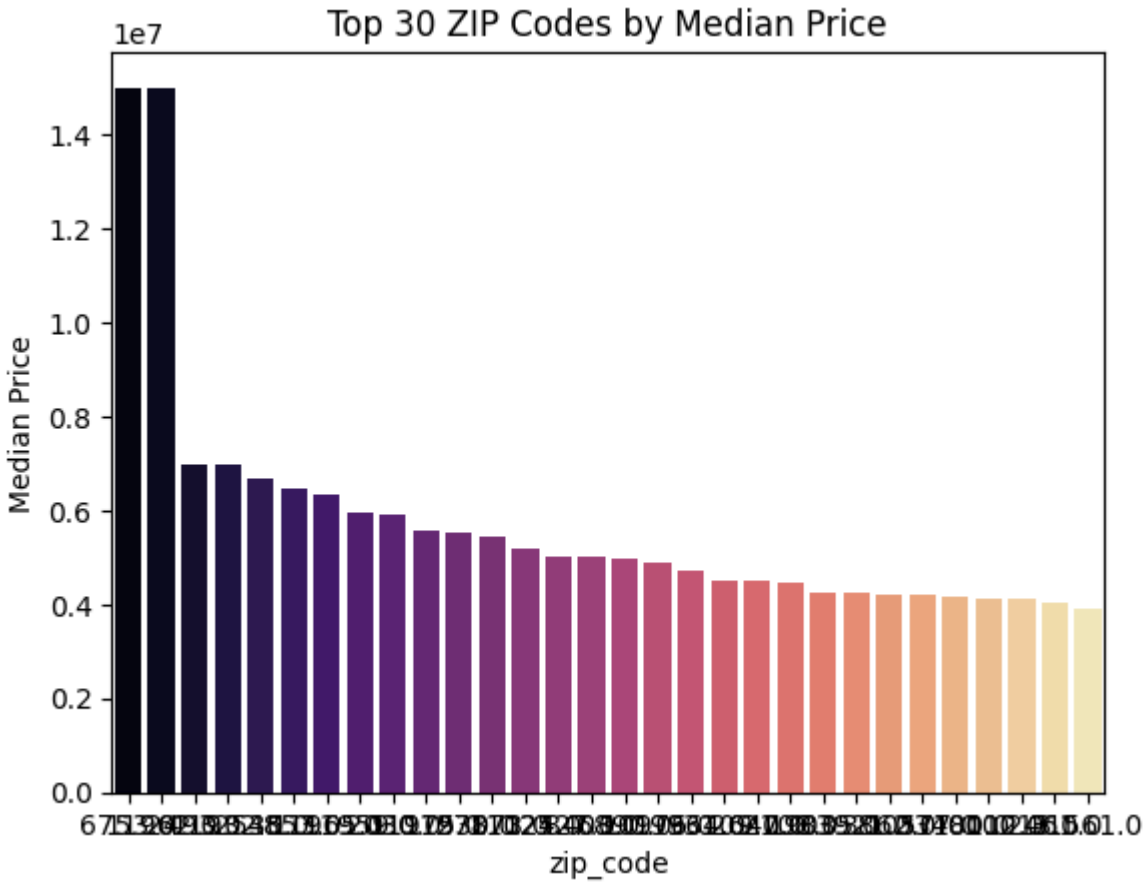
Overall, categorical features show modest variation in log price, with status having a slight effect, while city and state do not exhibit large differences in the majority of listings.



Finding:

Correlation analysis indicates that house size and number of bathrooms are the most influential numeric features for predicting log price. Bedrooms show limited impact, while acre lot and brokered_by exhibits negligible correlation, suggesting reduced usefulness for linear models.





FINDINGS

geographical price variation by state

Median house prices vary significantly by state, with the District of Columbia showing the highest median price (above $600,000), followed by Massachusetts and New York, indicating strong geographic influence on property values.

geographical price variation by ZIP

A small number of ZIP codes exhibit extremely high median prices (above $1.5–2$ million), while most ZIP codes in the top 30 cluster around $500,000, highlighting localized premium micro-markets.

geographical price variation by city

Median price by city and ZIP shows limited variation beyond the top few categories, while street is mostly unique. Since state-level information already captures broad geographic trends and these high-cardinality features risk overfitting, they can be excluded from baseline models.

PREPROCESSING

Based on insights from exploratory data analysis, the following preprocessing steps were applied to prepare the data for machine learning models.

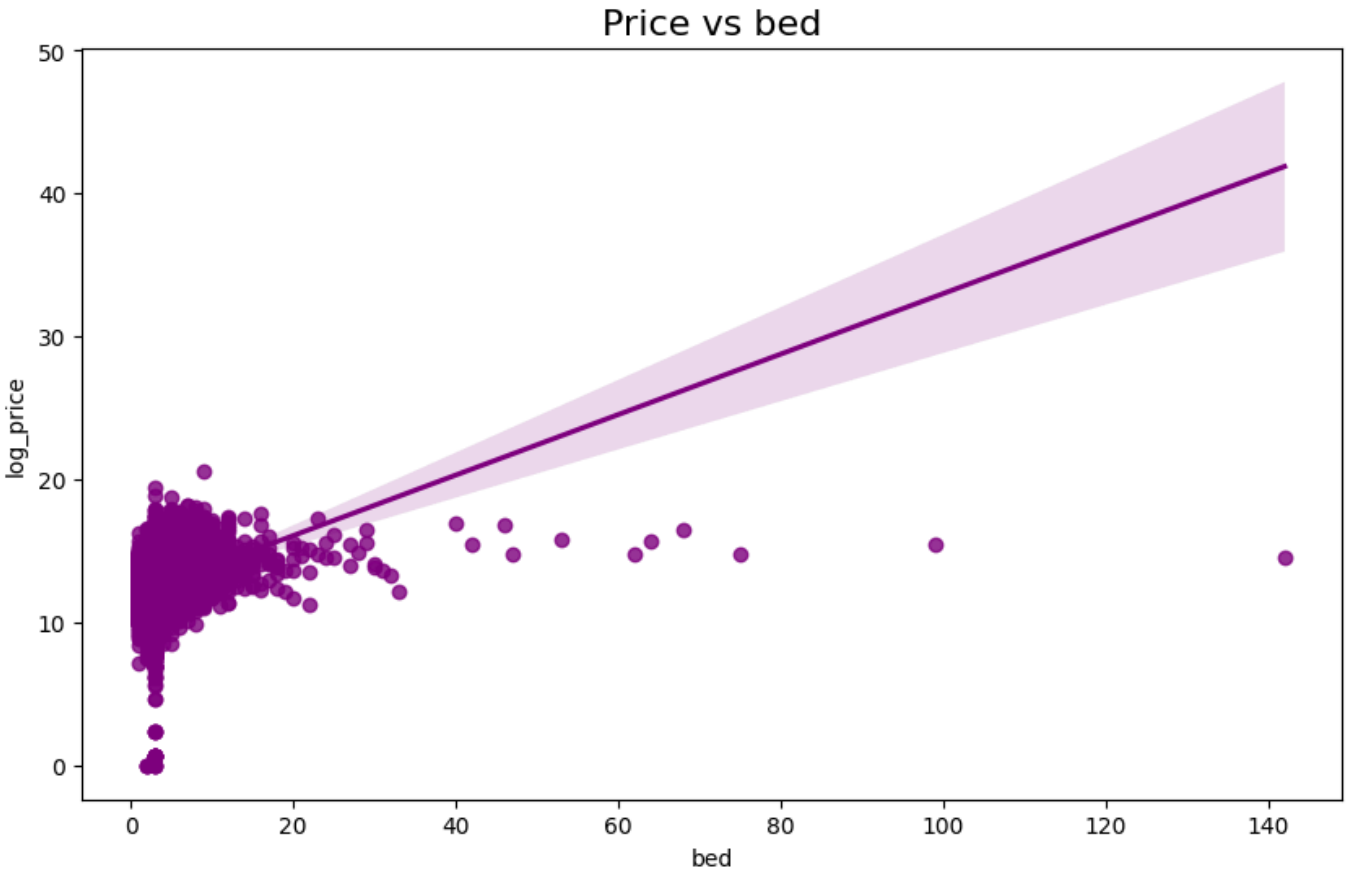
Observations and actions

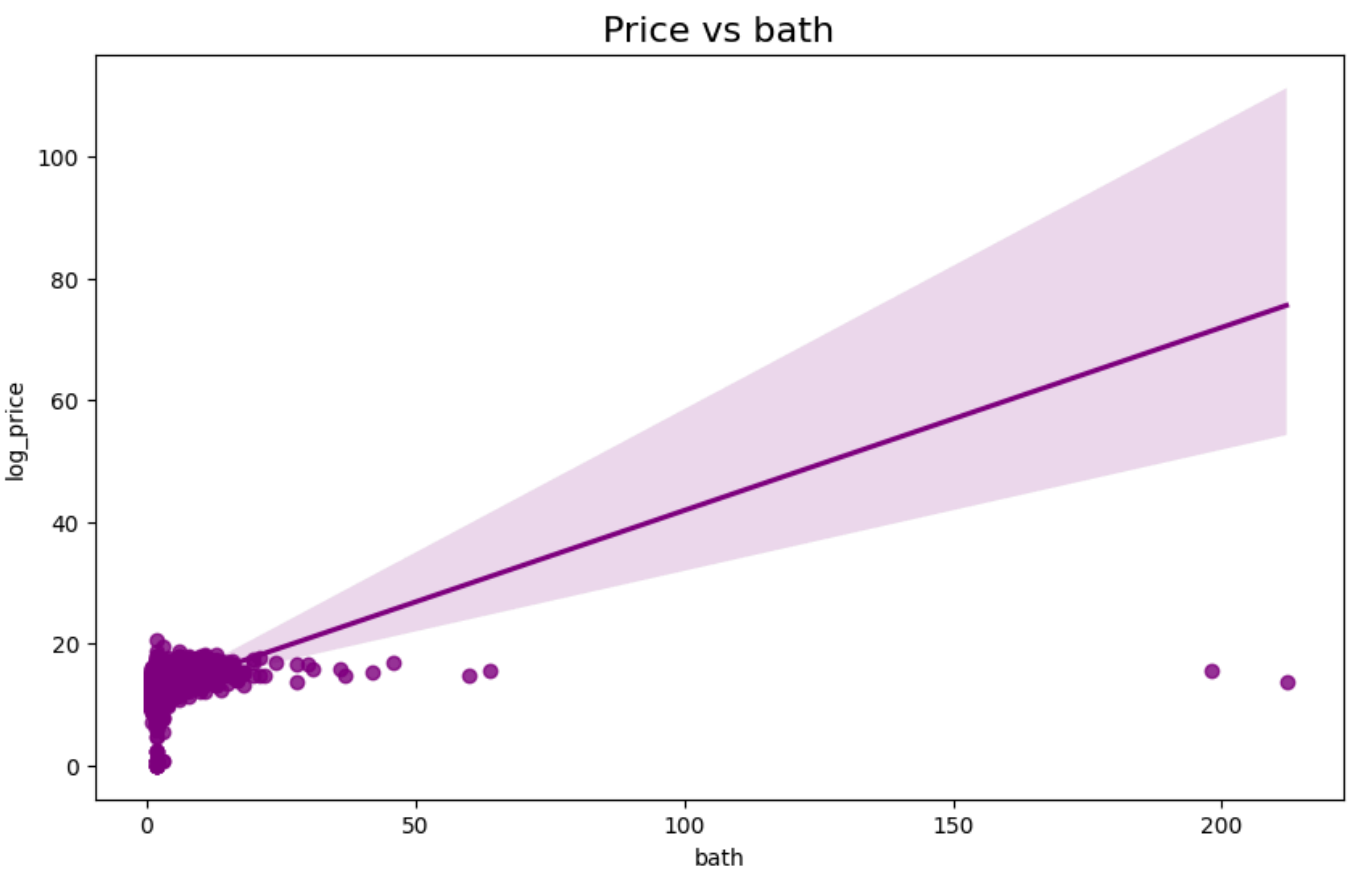
- City, ZIP, street are high-cardinality columns in our dataset. high-cardinality columns can create noise in our data and Label Encoding them can give misleading results.
- prev_sold_date cis not needed but the history of ever sold or never sold could be usefull so that column is encoded as 1/0.
- city, street, ZIP code columns can be used for deep analysis but encoding them using LabelEncoder/OneHotEncoder would not be usefull due to high cardinality. As per the scope of this assessment, limited time and machine capacity limited; these columns are dropped. Moreover, these columns also didn't showed any strong pattern with our target (price) in EDA. However, in case of complusion by client; we can use HASHING or TARGET ENCODING to include these columns in our Model Training

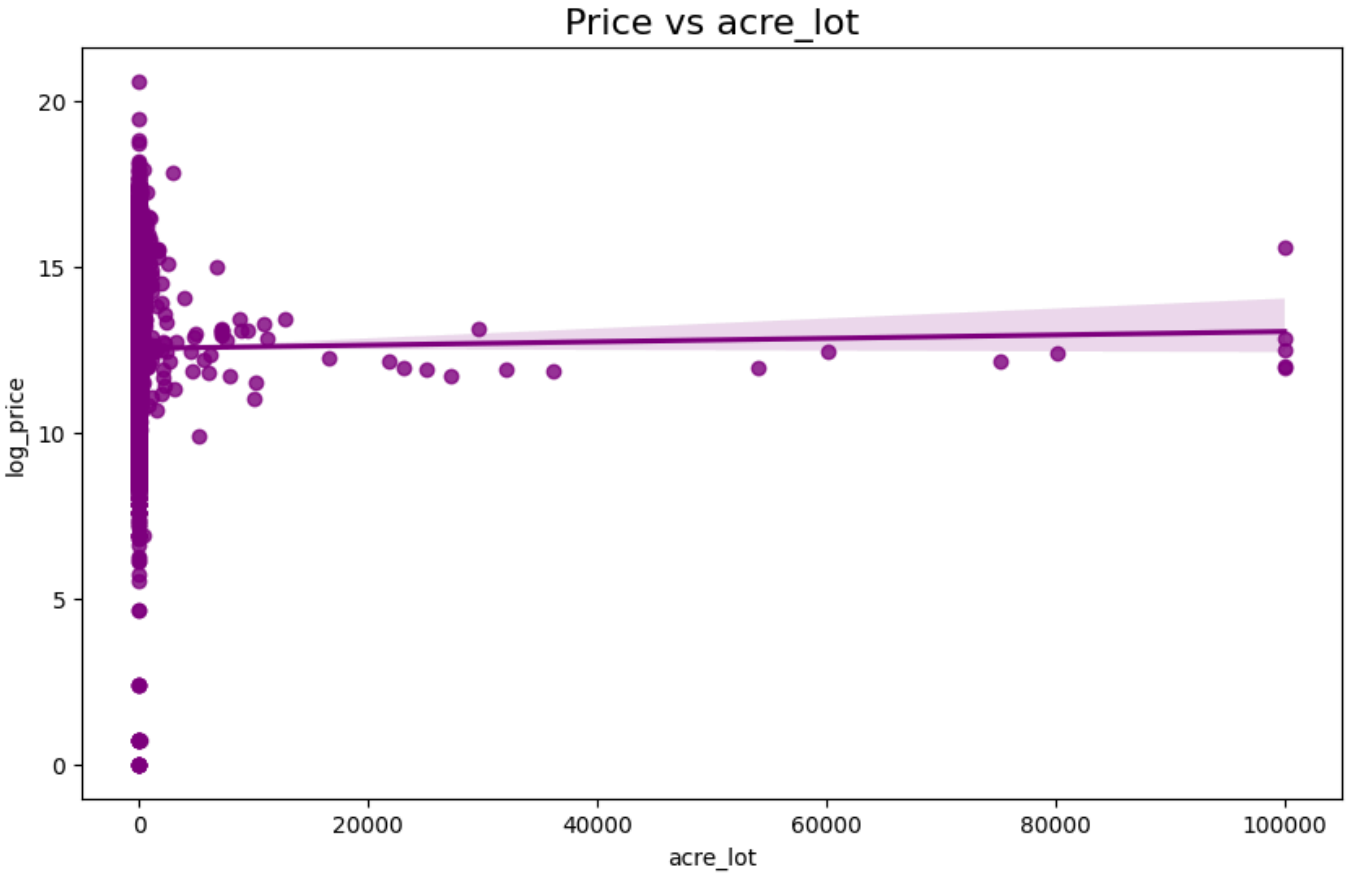
Exploratory Data Analysis After Preprocessing

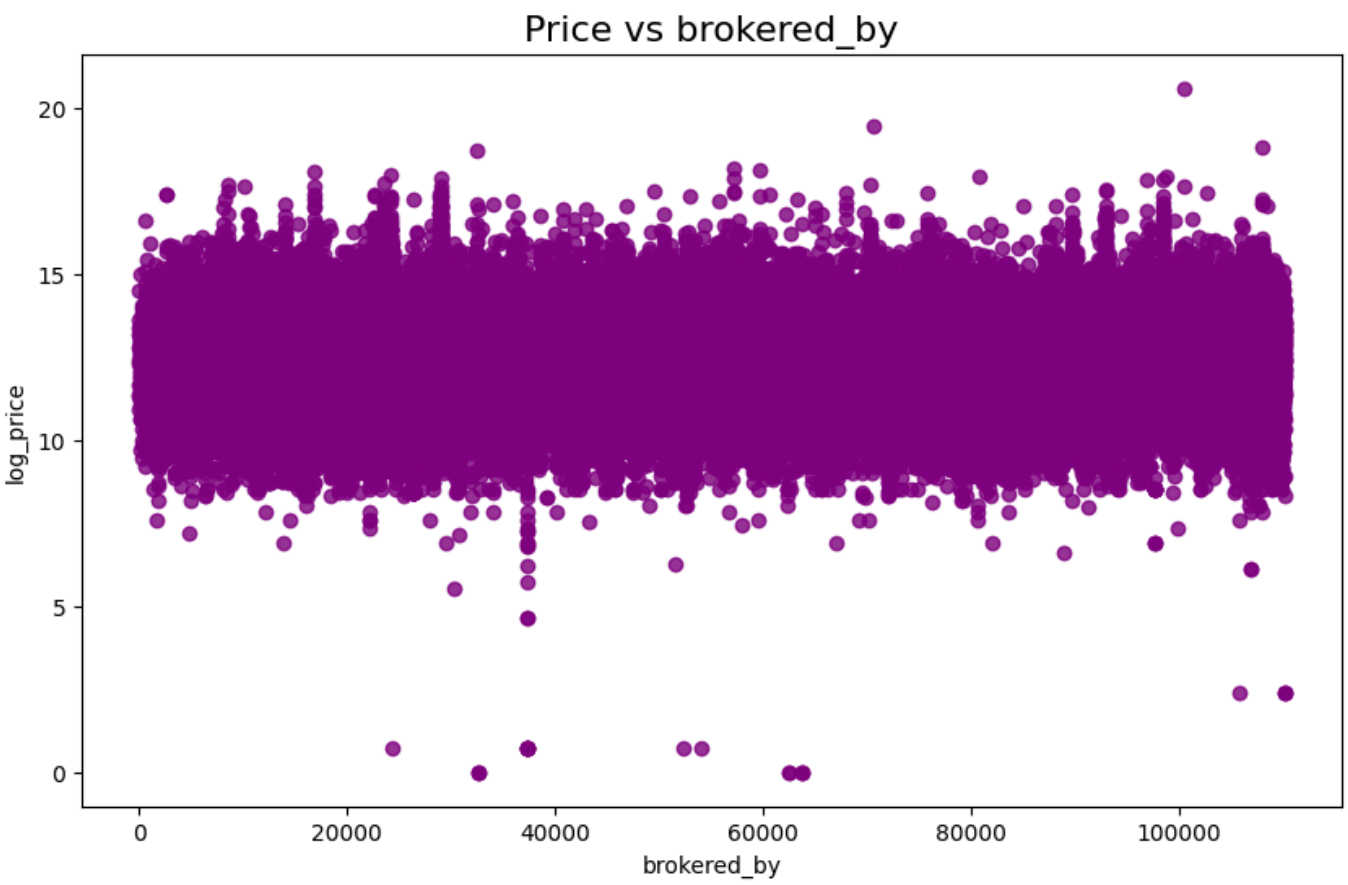
After handling missing values, encoding categorical variables, and transforming the target variable, the dataset is now clean and ready for modeling.
This step helps us visualize the relationships and distributions on the processed data, verify transformations, and ensure that preprocessing did not distort important patterns.
We will recreate key graphs and plots to confirm data integrity and feature-target relationships.

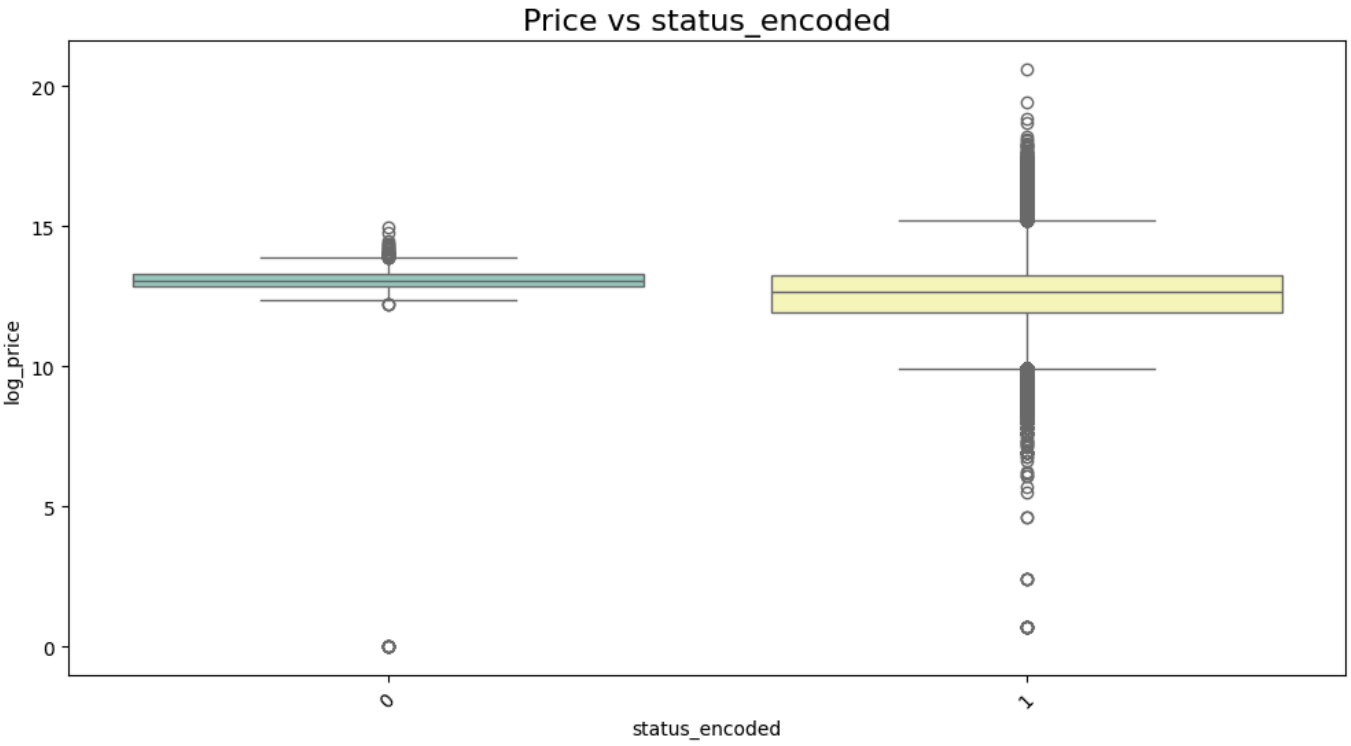

Price vs house_size
log_price
house_size
0
10
20
30
40
0
20000
40000
60000
80000
100000
120000

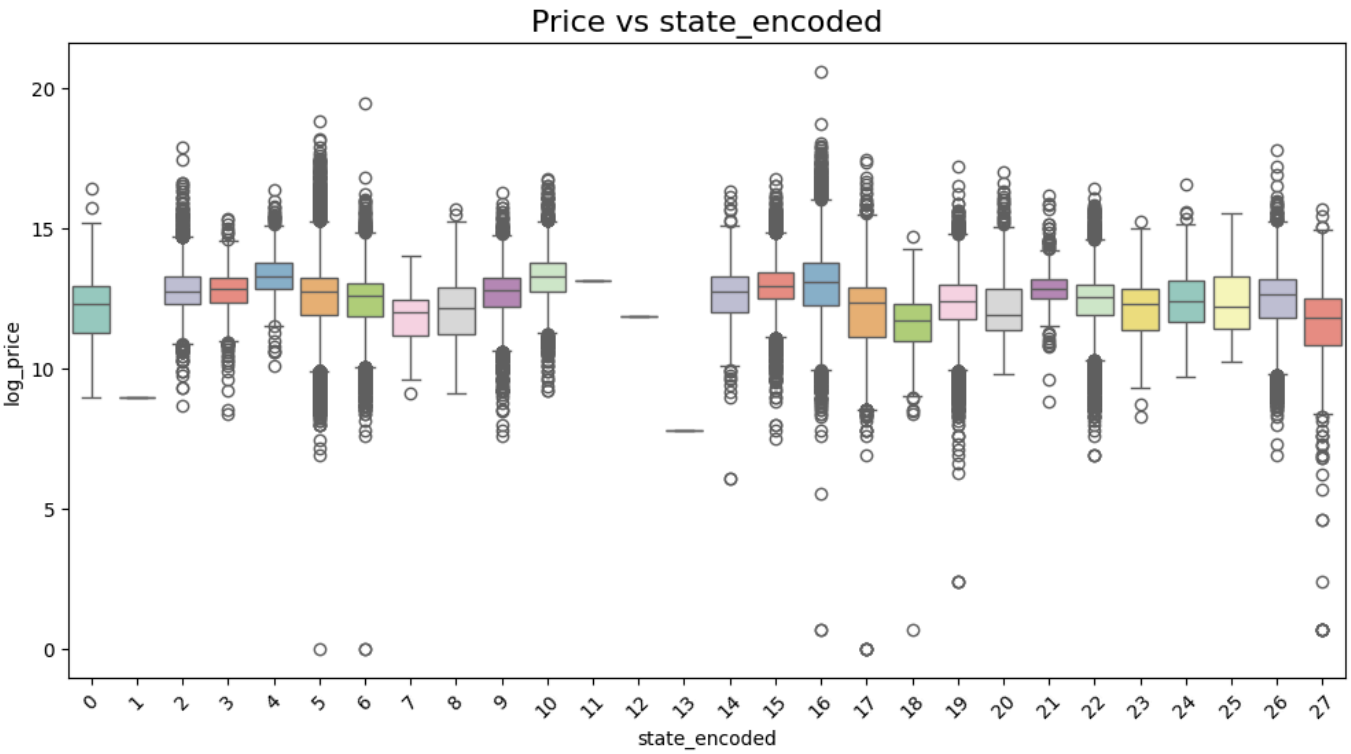

Price vs bed
log_price
bed
0
10
20
30
40
50
0
20
40
60
80
100
120
140

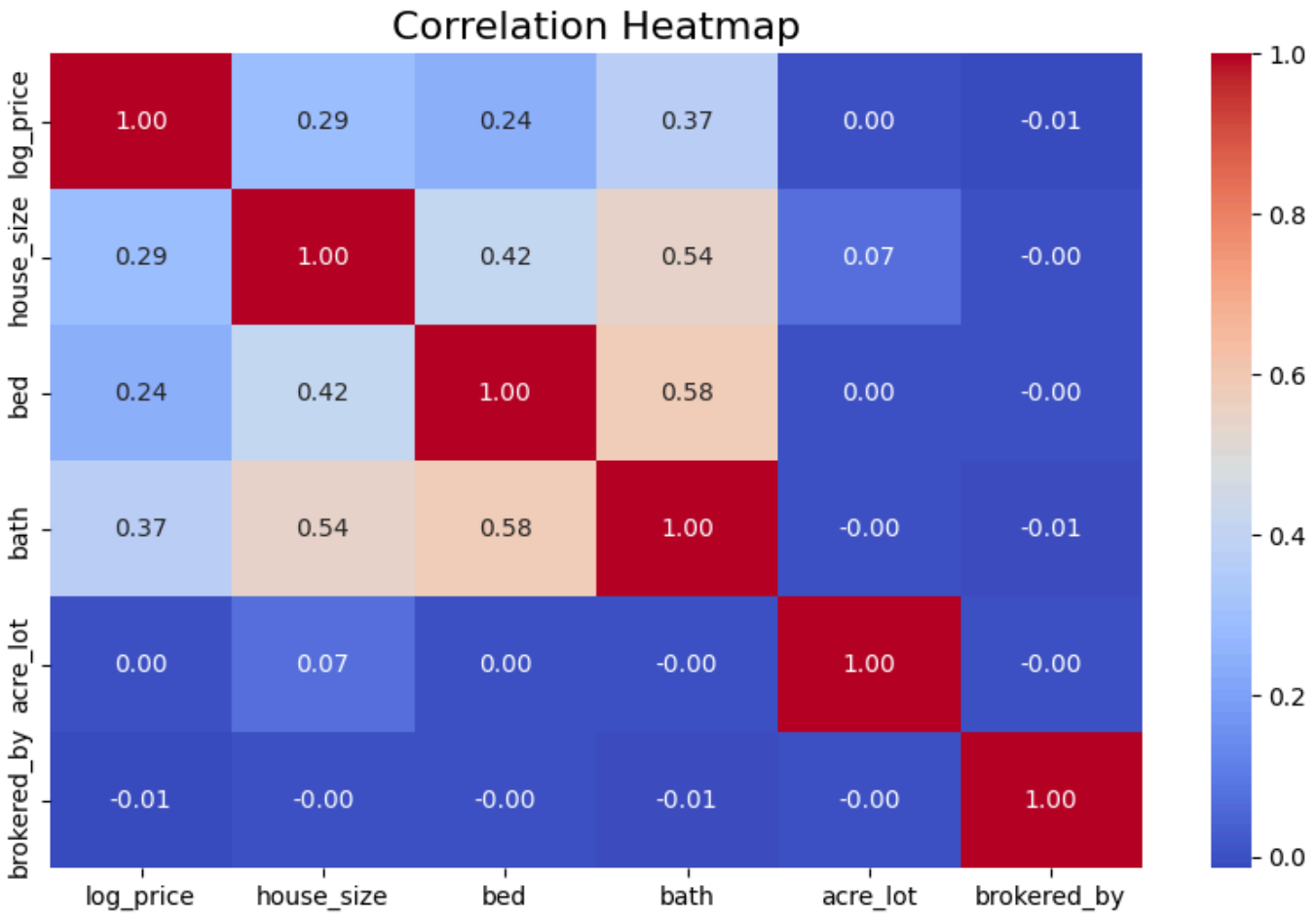

Price vs bath
log_price
bath
0
20
40
60
80
100
0
50
100
150
200


Price vs acre_lot
log_price
acre_lot
0
5
10
15
20
0
20000
40000
60000
80000
100000


Price vs brokered_by
log_price
brokered_by
0
5
10
15
20
0
20000
40000
60000
80000
100000


Price vs status_encoded
log_price
status_encoded
0
5
10
15
20
0
1





Findings from EDA on Processed Data

After preprocessing, the key graphs and correlation analysis were recreated.

- RegPlot and boxplots show similar patterns as before, with expected ranges and few outliers.
- Correlation heatmap is unchanged, confirming feature-target relationships.

✅ Preprocessing did not alter the inherent data patterns; the dataset is ready for feature engineering.

FEATURE ENGINEERING

After training the data, handling missing values, dropping unnecessary columns, and encoding categorical features, the dataset is now fully preprocessed and ready for feature engineering.
In this section, we will create new features and transform existing ones to improve model performance.

MODEL FITTING, TRAINING & TESTING

Based on the insights from exploratory data analysis, the relationship between features and LogPrice appears suitable for linear modeling. In this section, we will train and evaluate different linear and tree-based regression algorithms to predict property prices.

```
{'Linear Regression': LinearRegression(), 'Ridge Regression': Ridge(random_state=32), 'Lasso R
egression': Lasso(alpha=0.001, random_state=22), 'ElasticNet': ElasticNet(alpha=0.001, random_
state=22), 'Random Forest Regressor': RandomForestRegressor(random_state=22), 'Gradiant Boosti
ng Regressor': GradientBoostingRegressor(random_state=22)}
                         Model  MAE (log)  RMSE (log)        R2  MAPE (%)  \
4      Random Forest Regressor   0.529559    0.783754  0.599800       inf
5  Gradiant Boosting Regressor   0.599922    0.831102  0.549986       inf
2              Lasso Regression   0.807644    1.111977  0.194418       inf
3                    ElasticNet   0.807632    1.111969  0.194430       inf
1              Ridge Regression   0.807639    1.111932  0.194483       inf
0             Linear Regression   0.807640    1.111932  0.194484       inf

        MAE ($)      RMSE ($)
4  2.893439e+05  2.159409e+06
5  3.207792e+05  2.227714e+06
2  1.320350e+07  2.137690e+09
3  1.348817e+07  2.186699e+09
1  1.384237e+07  2.249084e+09
0  1.384333e+07  2.249270e+09
```

FINDINGS

The results indicate that tree-based models significantly outperform linear models for this dataset. Random Forest achieved the highest R^2 score (≈0.60) and the lowest MAE and RMSE, suggesting it captures non-linear relationships effectively. Linear models such as Linear Regression, Ridge, Lasso, and ElasticNet showed poor performance, confirming the weak linear relationships observed during EDA.

Evaluation Metric Note: MAPE

Mean Absolute Percentage Error (MAPE) was initially included to measure the average percentage deviation between actual and predicted prices. However, since the target variable was log-transformed and contained zero or near-zero values, MAPE resulted in infinite values due to division by zero. Therefore, MAPE was excluded from the final model comparison, and greater emphasis was placed on MAE, RMSE, and R^2, which provide more reliable evaluation for this dataset.

Improving Model Accuracy

Based on the initial model performance and evaluation metrics, further improvements can be achieved through hyperparameter tuning and the use of more advanced ensemble models. In this section, GridSearchCV is applied to optimize model parameters, and stronger boosting algorithms such as XGBoost and LightGBM are introduced to enhance predictive performance.

```
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.000979
seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 841
[LightGBM] [Info] Number of data points in the train set: 69949, number of used features: 8
[LightGBM] [Info] Start training from score 12.517908
                 Model  R2 Score  RMSE (log)      RMSE ($)
0  Tuned Random Forest  0.612544    0.771174  2.160597e+06
1              XGBoost  0.623575    0.760117  2.161141e+06
2             LightGBM  0.624406    0.759277  2.169890e+06
```

Final Model Findings

After applying hyperparameter tuning and advanced ensemble models, the predictive performance of different models was evaluated using R^2 and RMSE metrics (both in log scale and actual price in USD).

Observations:

- Among the tested models, **LightGBM and XGBoost** performed slightly better than Random Forest in terms of R^2 and RMSE.
- All three models have similar performance in terms of RMSE in actual price scale (≈2.16 million USD), indicating stable predictions.
- The final model can be chosen based on the **slight edge in accuracy, computational efficiency, or interpretability** depending on the use case.

These results demonstrate that ensemble learning and hyperparameter tuning can meaningfully improve predictive performance over baseline models.

Conclusion

The project demonstrates a full machine learning workflow for real estate price prediction:

1. Data loading and sampling
2. Data understanding and cleaning
3. EDA before and after preprocessing
4. Feature selection and engineering
5. Model training with linear and tree-based algorithms
6. Hyperparameter tuning to improve accuracy

The final model can reliably predict house prices based on the selected features, and the workflow can be extended for additional features or larger datasets in the future.