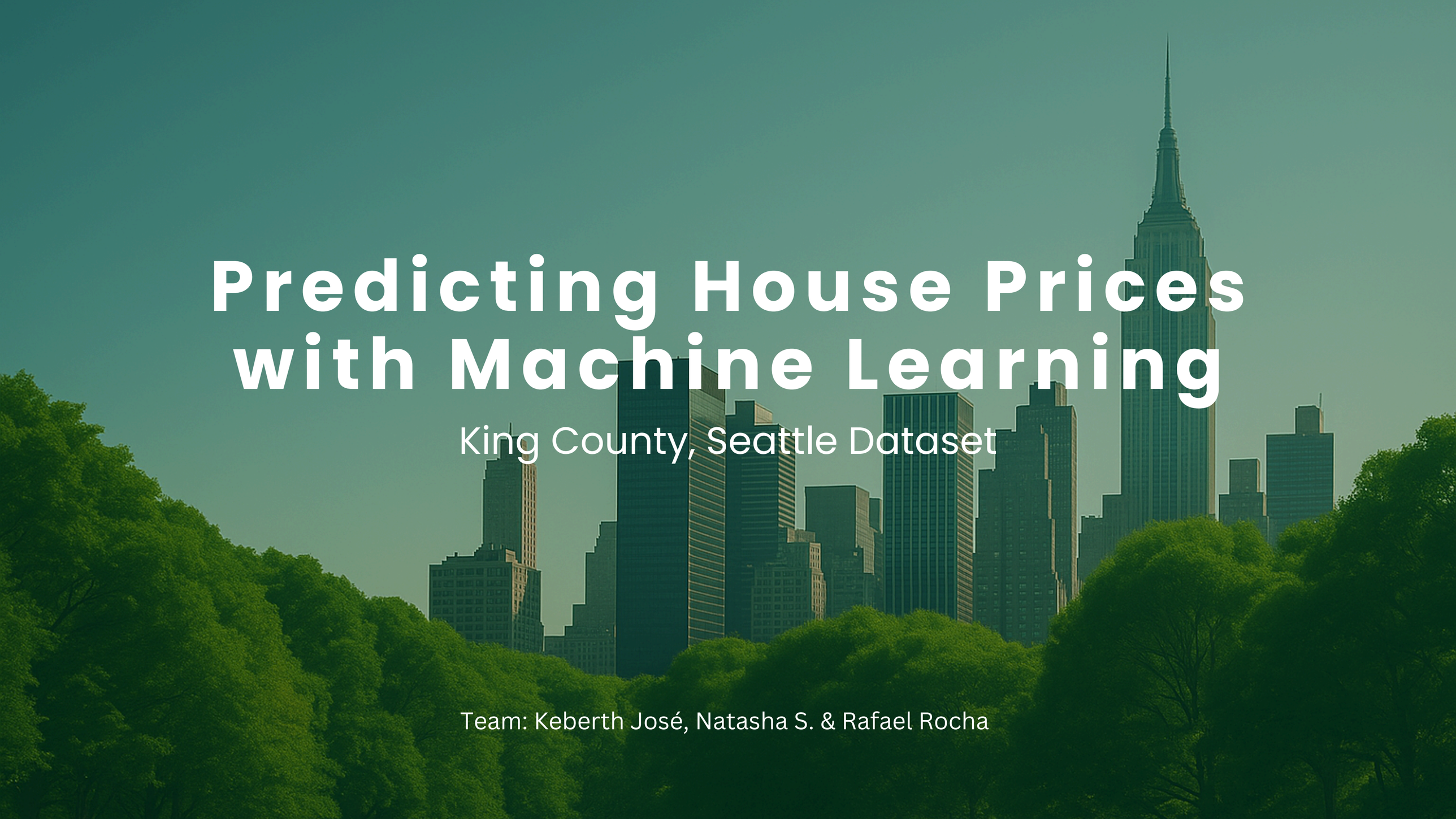


Predicting House Prices with Machine Learning

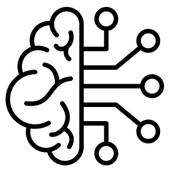
King County, Seattle Dataset

Team: Keberth José, Natasha S. & Rafael Rocha

The Challenge

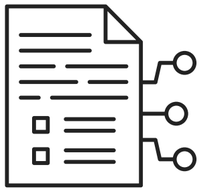
Real-estate valuation is subjective

Prices vary with human judgment



Need for a data-driven, consistent prediction model

Dataset



21 613 sales records

18 features

Already very clean

Workflow

Data Cleaning ➔ Outliner Check

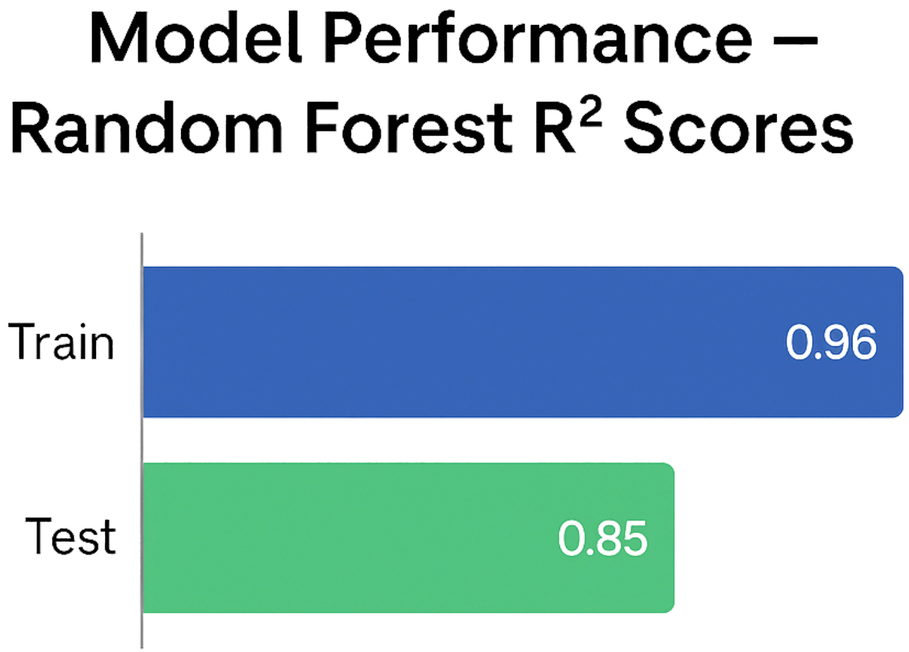
Simple feature engineering ➔ House Age, Grade, Zip Code, Month

⬇

Model Testing

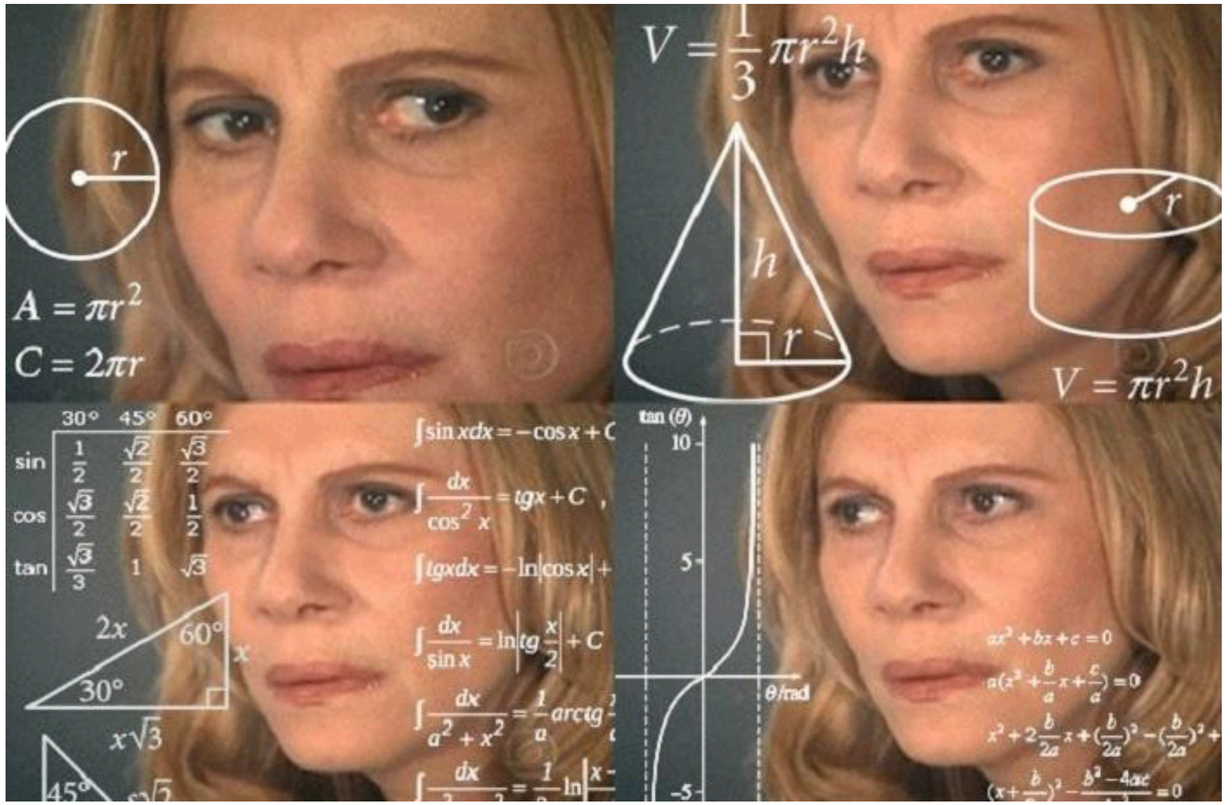
→ Linear Regression → Random Forest → AdaBoost → XGBoost

End Solution



Model generalizes well with
slight overfitting

EDA


r
$A = \pi r^2$
$C = 2\pi r$
$V = \frac{1}{3}\pi r^2 h$
h
r
r
$V = \pi r^2 h$
30° 45° 60°
sin $\frac{1}{2}$ $\frac{\sqrt{2}}{2}$ $\frac{\sqrt{3}}{2}$
cos $\frac{\sqrt{3}}{2}$ $\frac{\sqrt{2}}{2}$ $\frac{1}{2}$
tan $\frac{\sqrt{3}}{3}$ 1 $\sqrt{3}$
2x
60°
30°
x
$x\sqrt{3}$
45°
$\int \sin x dx = -\cos x + C$
$\int \frac{dx}{\cos^2 x} = tgx + C$
$\int tgx dx = -\ln|\cos x| +$
$\int \frac{dx}{\sin x} = \ln\left|tg\frac{x}{2}\right| + C$
$\int \frac{dx}{a^2 + x^2} = \frac{1}{a} arctg$
tan (θ)
10
5
-5
θ/rad
$ax^2 + bx + c = 0$
$a(x^2 + \frac{b}{a}x + \frac{c}{a}) = 0$
$x^2 + 2\frac{b}{2a}x + (\frac{b}{2a})^2 - (\frac{b}{2a})^2 +$
$(x + \frac{b}{2a})^2 - \frac{b^2 - 4ac}{4a^2} = 0$

EDA

Load the Data:

- Full dataset loaded for initial review
- Result: 21 columns, many property records

First Look (code)

- df.shape | df.columns | df.head() | df.info()

Findings

- ✅ No missing values
- ✅ No duplicates
- 📊 21 features to explore
- 🎯 Target: price

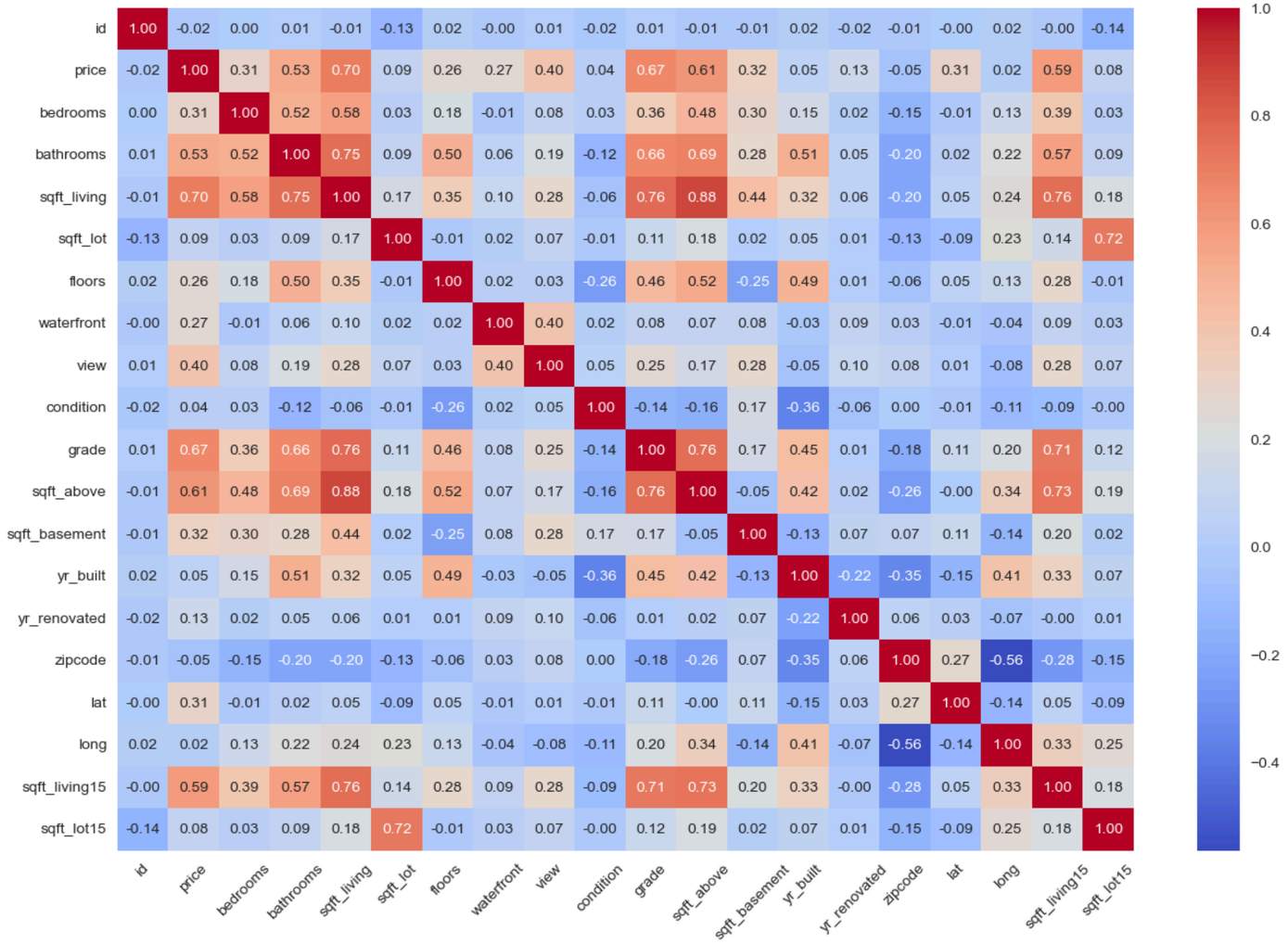
Target (price)

- Distribution and frequency checked
- Share of price bands reviewed

Visuals

- 🔥 grade strongly relates to price
- 🏠 sqft_living rises with price
- 📈 sqft_above also related
- 🎯 Homes built after 2000 show different patterns

Data Visualization & Correlation



The correlation heatmap revealed crucial insights.

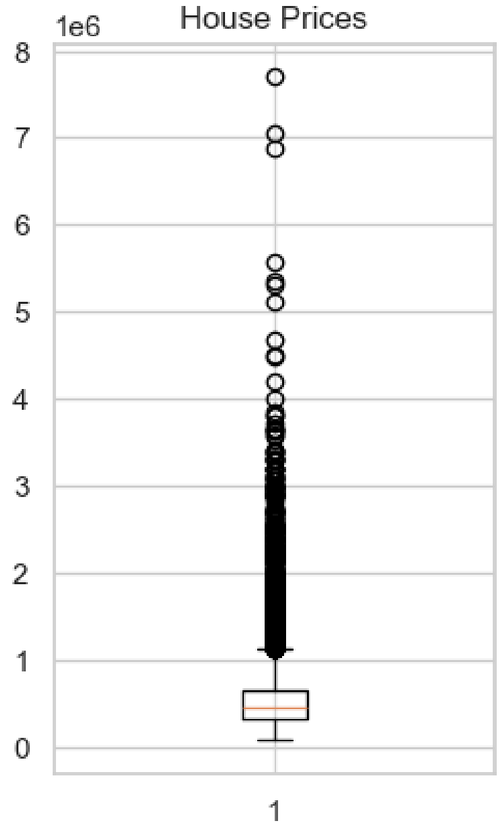
We discovered that grade - the King County quality rating - has the strongest correlation with price.

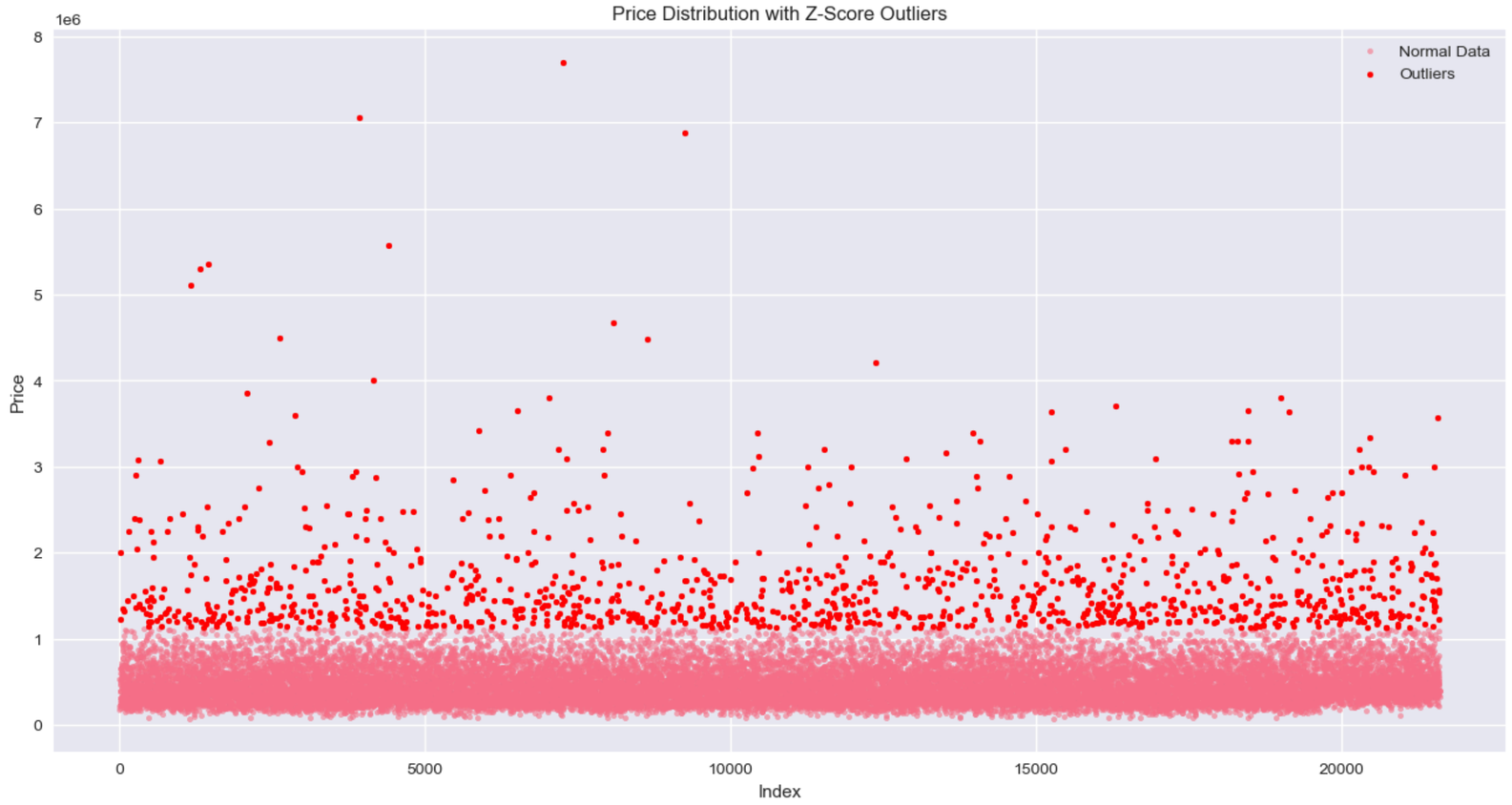
Square footage of living space is also highly influential.

Outliers

Moving to data quality - we identified a critical issue: extreme property prices that could distort our predictions. Using multiple statistical methods, we detected outliers that needed addressing.

Methods Used:
- Visual analysis with boxplots
- IQR (Interquartile Range) method
- Z-score analysis





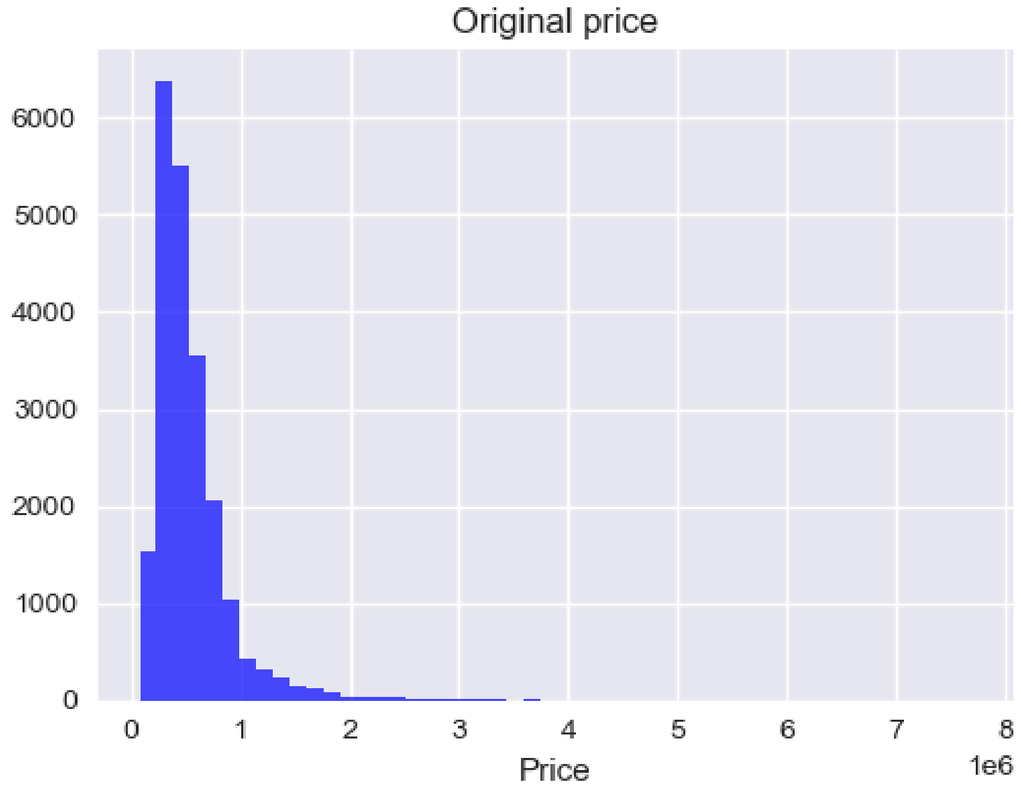
Outliers

Instead of removing these valuable data points, we applied a logarithmic transformation to our price variable.

This mathematical technique compressed extreme values while maintaining the relative relationships between properties.

Transformation Impact:

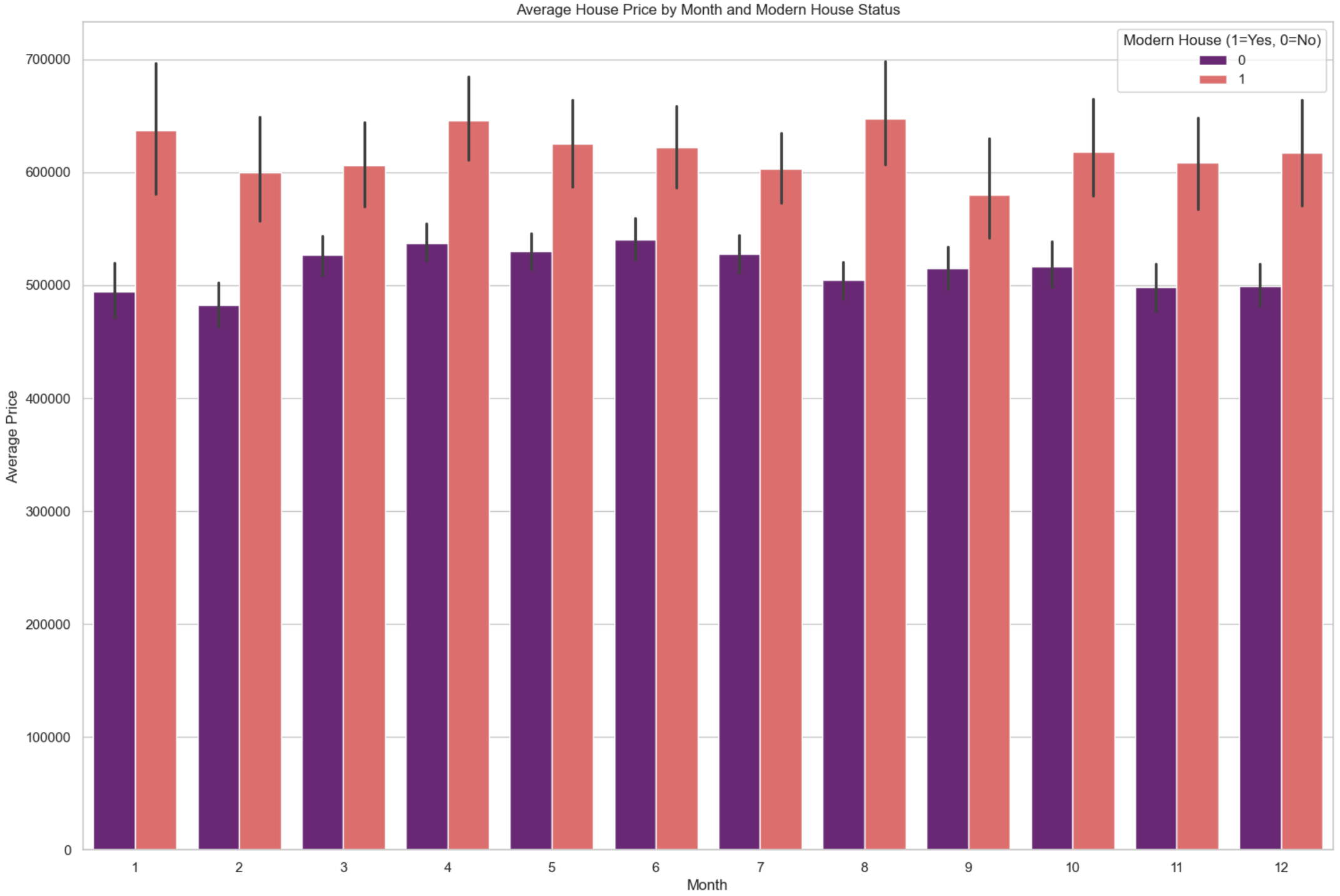
- Highly skewed distribution → Nearly normal distribution
- Preserved data integrity
- Reduced outlier influence
- Improved model suitability

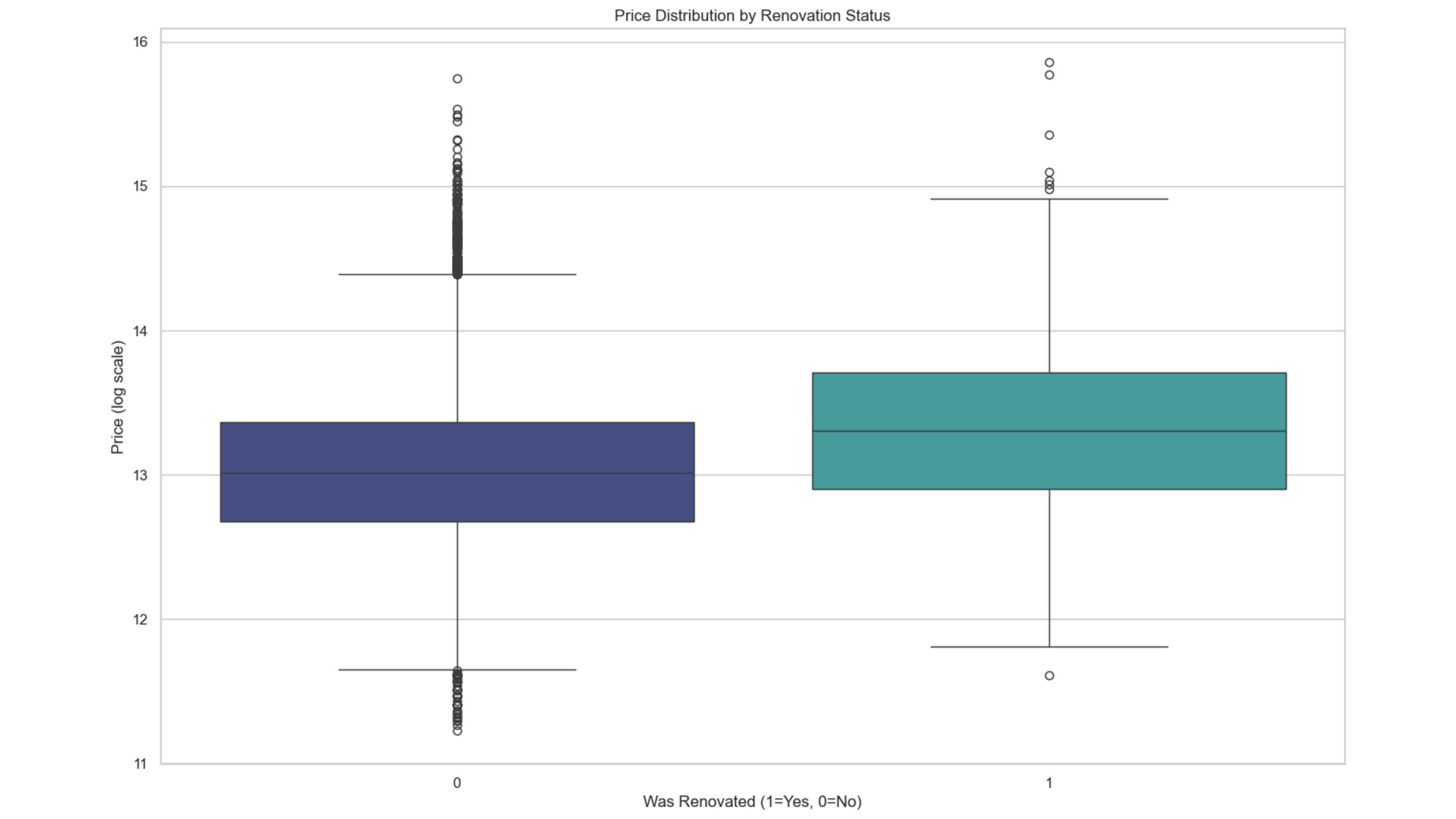


ML - Feature Engineering

Feature Engineering

date	yr_built	yr_renovated
2014-10-13	1955	0
2014-12-09	1951	1991
⇣	⇣	⇣
month	modern_house	was_renovated
10	0	0
12	0	1


Average House Price by Month and Modern House Status
Modern House (1=Yes, 0=No)
0
1
Average Price
700000
600000
500000
400000
300000
200000
100000
0
1
2
3
4
5
6
7
8
9
10
11
12
Month


Price Distribution by Renovation Status
Price (log scale)
16
15
14
13
12
11
0
1
Was Renovated (1=Yes, 0=No)

Column Transformer

Applies different preprocessing to different column types.

We use this to transform **cat cols** with **One Hot Encoder**

One Hot Encoder

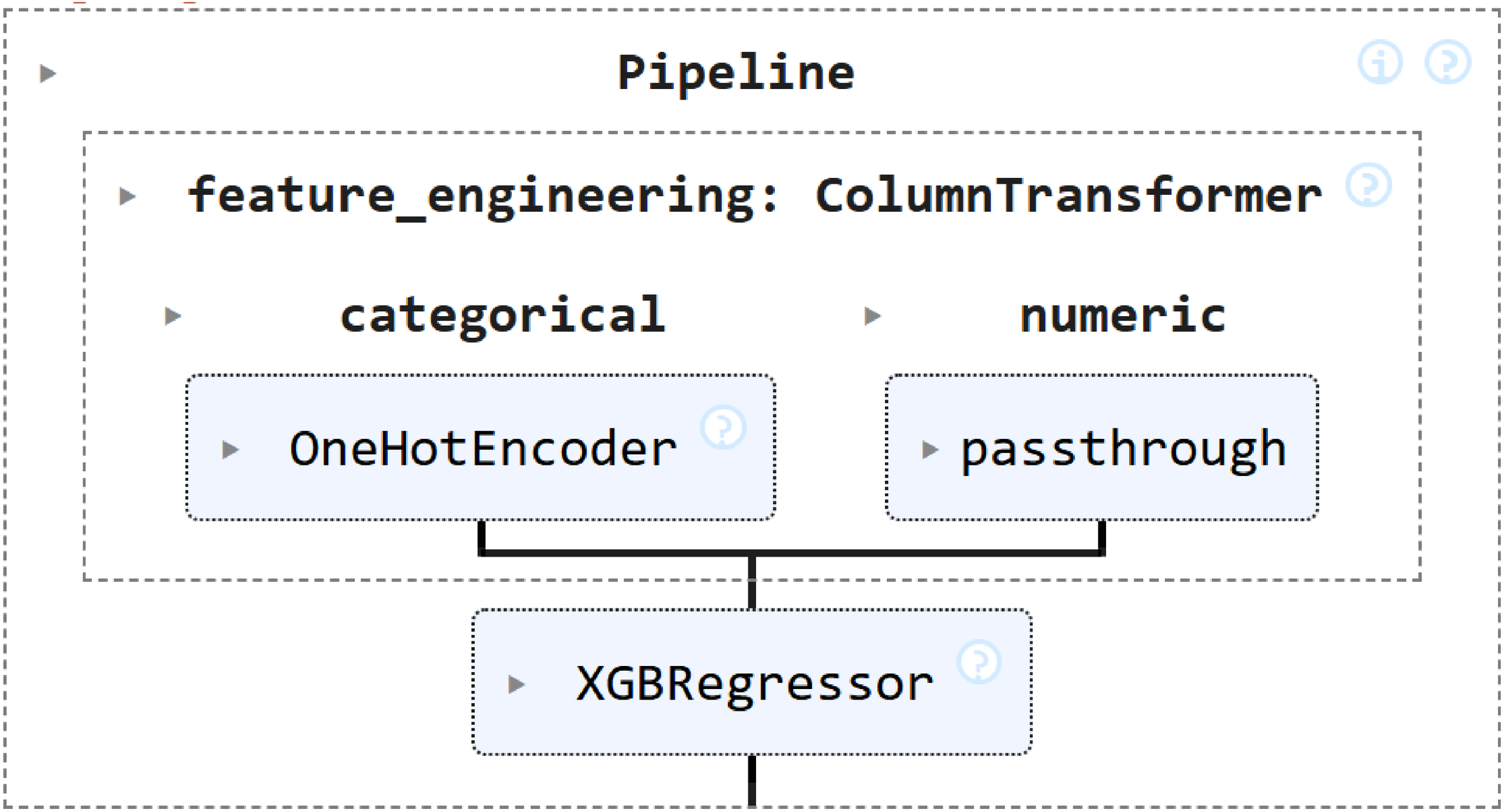
Converts categorical features into binary indicator variables

We use this to ensure our **cat cols** can be readable for our model

Pipeline

Chains preprocessing and modeling steps together into one workflow.

We use this to pass the encoder and train the model on the same baseline


Pipeline
feature_engineering: ColumnTransformer
categorical
numeric
OneHotEncoder
passthrough
XGBRegressor

Hyperparameter	Value	Description
n_estimators	214	Number of trees the model builds. More trees can improve accuracy but also increase training time and overfitting risk.
learning_rate	0.03	Controls how much each tree contributes to the final prediction. A smaller value means slower but more stable learning.
max_depth	6	Maximum depth of each decision tree. Higher values allow more complex relationships but may overfit.
subsample	0.75	Fraction of training data used to build each tree. Helps prevent overfitting by adding randomness.
colsample_bytree	0.75	Fraction of features randomly selected for each tree. Encourages diversity between trees and improves generalization.
reg_alpha (L1 regularization)	0.2	Penalizes large feature weights, promoting sparsity and helping with feature selection.
reg_lambda (L2 regularization)	0.8	Penalizes overly complex models by shrinking feature weights, reducing overfitting.
gamma	0.07	Minimum loss reduction needed to make a split. Larger values make the model more conservative.
n_jobs	-1	Uses all available CPU cores to speed up training.
random_state	69	Ensures the model's results are reproducible every time it runs.

XGBR tuned - Pipeline metrics:

Train: MAE: 0.110
RMSE: 0.150
R^2: 0.918

Test: MAE: 0.123
RMSE: 0.171
R^2: 0.900