Iron Kaggle Mini Project

Predicting House Prices Using Machine Learning Models

Team Ravenclaw:
Abhi, Debora, Emilia, & Zhoubin
21.03.2025

Focus of Analysis

- **Primary Objective**
  - Experimenting with different models to predict house prices
- **Secondary Focus**
  - Explore properties valued **$650K and above** for deeper insights
- **Target Variable**
  - **`price`**: The sale price of the house is the central target measure

Key Features in the Dataset

- **Unique Identifier**
  - Unique ID for each house as **`id`**
- **Temporal Data**
  - Sale date of the house as **`date`**
- **Target Variable**
  - Sale price of the house (prediction target) as **`price`**
- **Property Characteristics**
  - **`bedrooms`**: Number of bedrooms
  - **`bathrooms`**: Number of bathrooms per bedroom
  - **`sqft_living`**: Interior living space (sq. ft.)
  - **`sqft_lot`**: Land space (sq. ft.)
  - **`floors`**: Number of floors
  - **`waterfront`**: Whether the house has a waterfront view
  - **`view`**: Number of times the house was viewed
  - **`condition`**: Overall condition of the house
  - **`grade`**: Overall grade based on King County grading system

- **Timeframe**: May 2014 to May 2015 (one-year data)
- **Location**: King County, including Seattle
- **Size**: 21 columns, 20K+ rows;

Additional Features

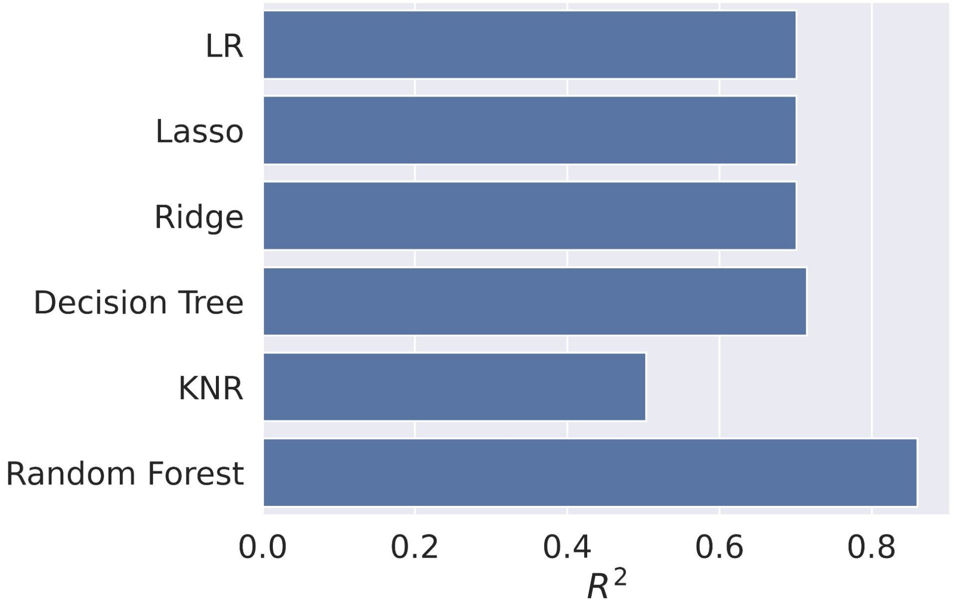
- **Structural Details**
  - `sqft_above`: Square footage excluding the basement
  - `sqft_basement`: Square footage of the basement
  - `yr_built`: Year the house was built
  - `yr_renovated`: Year the house was renovated
- **Location Data**
  - `zipcode`: ZIP code area
  - `lat`: Latitude coordinate
  - `long`: Longitude coordinate
- **Renovation Indicators**
  - `sqft_living15`: Living room area in 2015 (implies renovations)
  - `sqft_lot15`: Lot size area in 2015 (implies renovations)

EDA and Data Clean up

- 21613 rows and 21 columns
- 1 object (date) and 20 float/int columns
- Clean dataset by default (no NaN values, no empty spaces)
- High correlation (> 0.5) between **`price`** and the following features:
    - **`sqft_living`** `0.702`
    - **`grade`** `0.667`
    - **`sqft_above`** `0.606`
    - **`sqft_living15`** `0.585`
    - **`bathrooms`** `0.525`

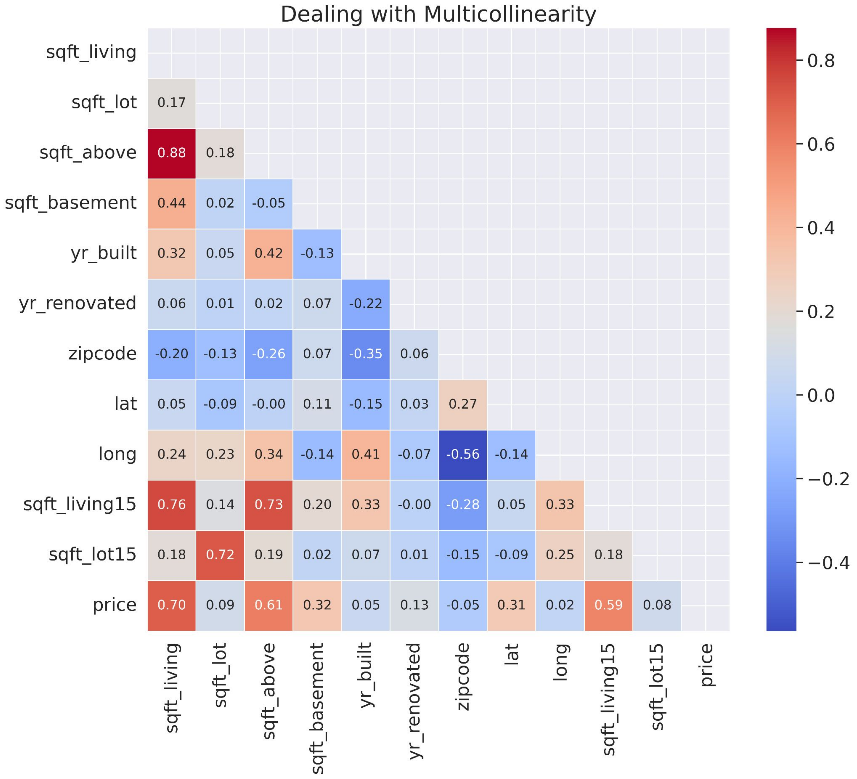
Baseline Models

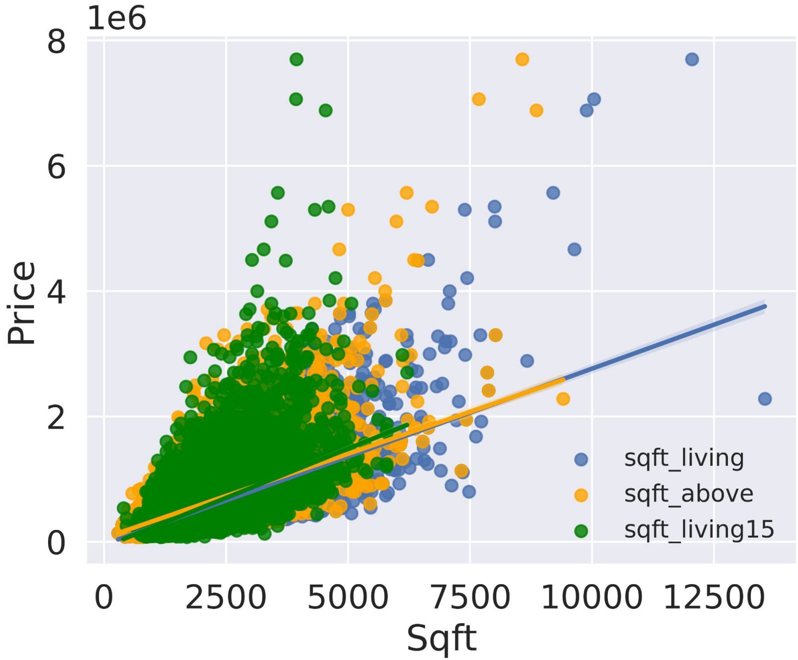
No feature engineering, only dropped: `columns=["id", "date"]`



How can we improve the models?

Multicollinearity



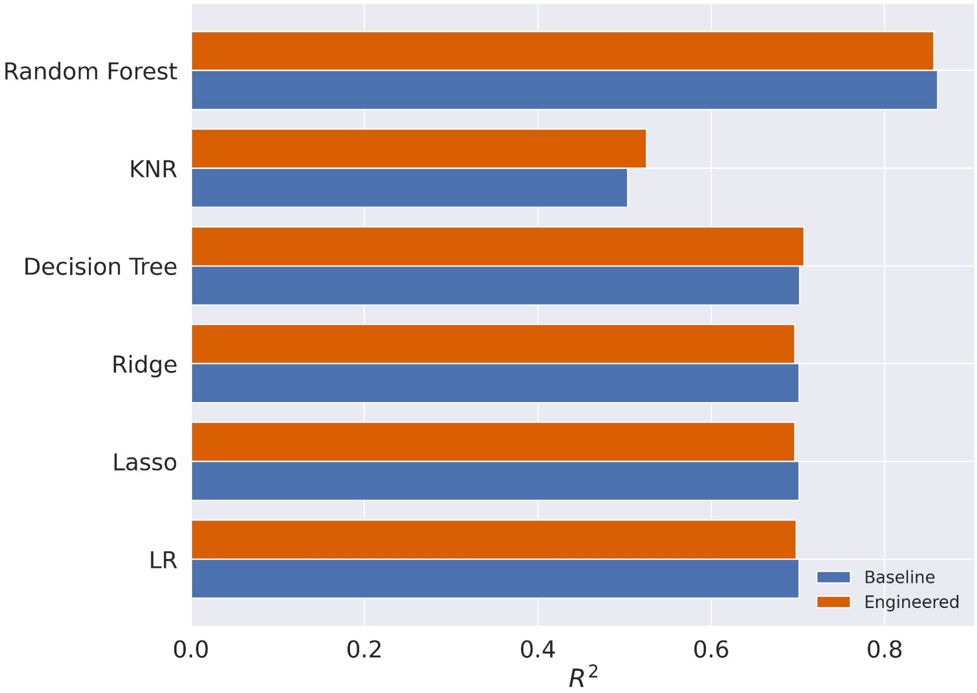


Feature Engineering Scenarios

- Removing the outliers using quartile measures
- Adding a house `age` feature and removing `yr_renovated`
    - `df["age"] = 2014 - df["yr_built"]`
    - `df.drop(columns=["yr_built", inplace=True)`
- Adding a binary feature `was_renovated` with 0/1 values
    - `df["was_renovated"] = (df["yr_renovated"] > 0).astype(int)`
    - `df.drop(columns=["yr_renovated", inplace=True)`
- Selecting a subset of **top 10** most influential features
    - ```
      features = ['sqft_living', 'grade', 'sqft_above', 'sqft_living15',
      'bathrooms', 'view', 'sqft_basement', 'bedrooms',
      'lat', 'waterfront']
      ```
- One-hot encode categorical features
    - `features = pd.get_dummies(features, columns=["zipcode"], drop_first=True)`
      ```

Feature Engineering

- **age**: 2014 - year_built
- **total_rooms** = bedrooms + bathrooms
- **is renovated or not**
- dropped: **sqft_above**, **sqft_living15**, **sqft_lot15**

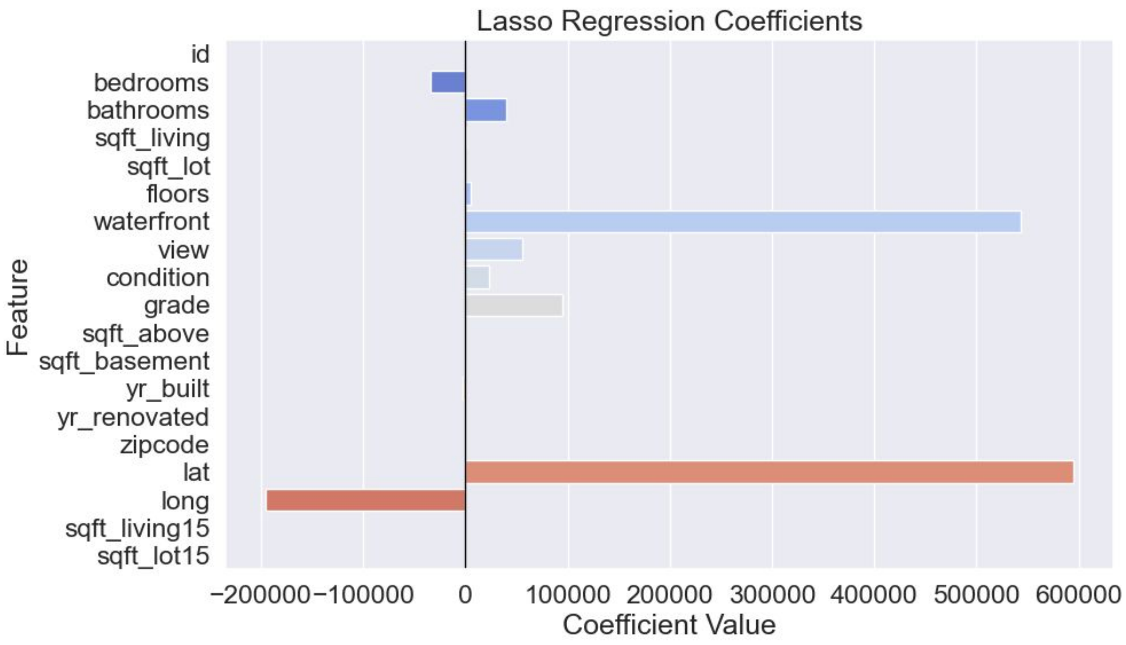


May be just train with numerical columns?

Benchmark of 7 Baseline Models

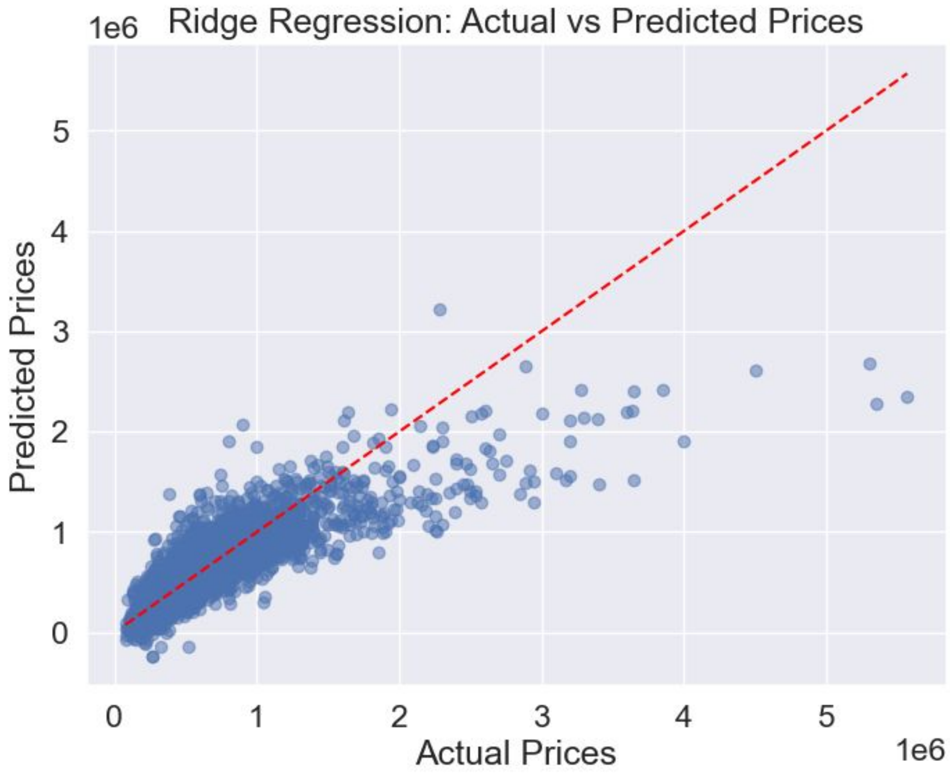
	MAE	RMSE	R^2
Linear Regression	127493.3	212539.5	0.701
Lasso Regression	127493.3	212539.6	0.701
Ridge Regression	127491.4	212540.1	0.701
Decision Tree	103550.5	206398.9	0.718
XGBoost	100812.0	190203.0	0.761
KNN Regressor	93170.4	182449.3	0.780
Random Forest	73092.7	148833.0	0.853

Lasso Regression



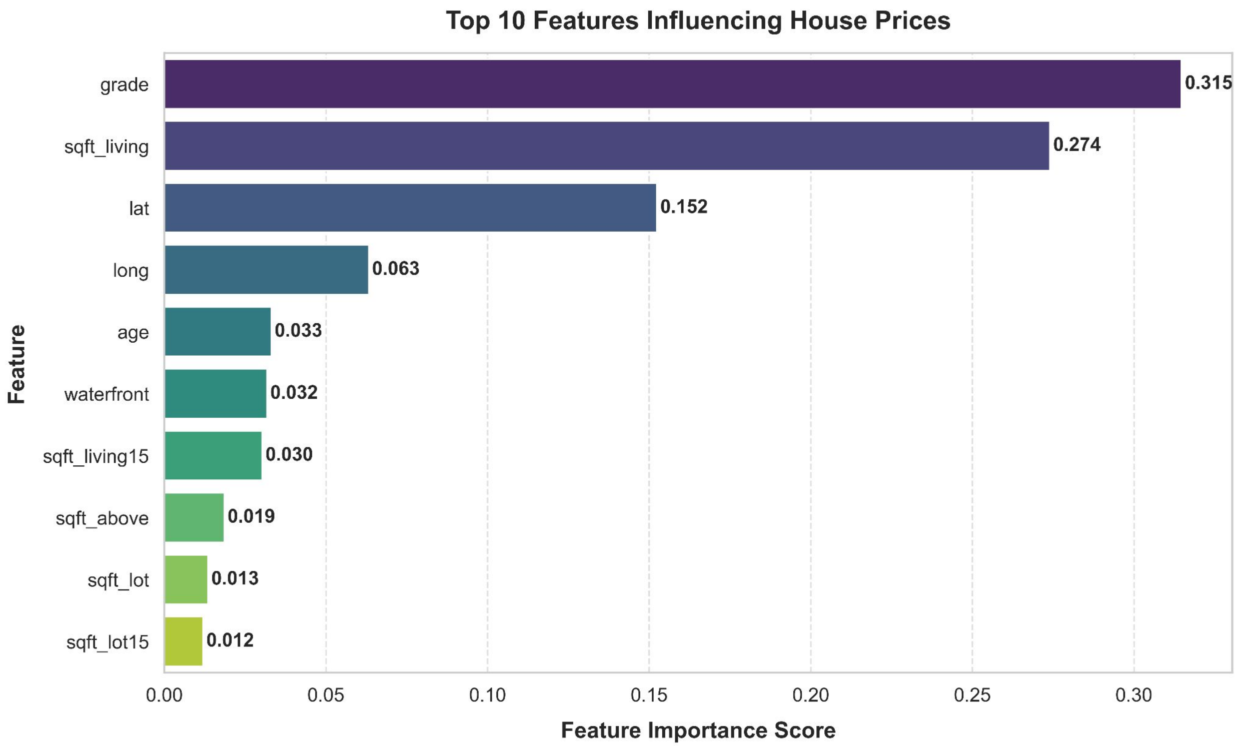
- Applied Lasso Regression to numerical variables initially, then included categorical variables.
- All variates contributed to predicting house prices.

Ridge Regression

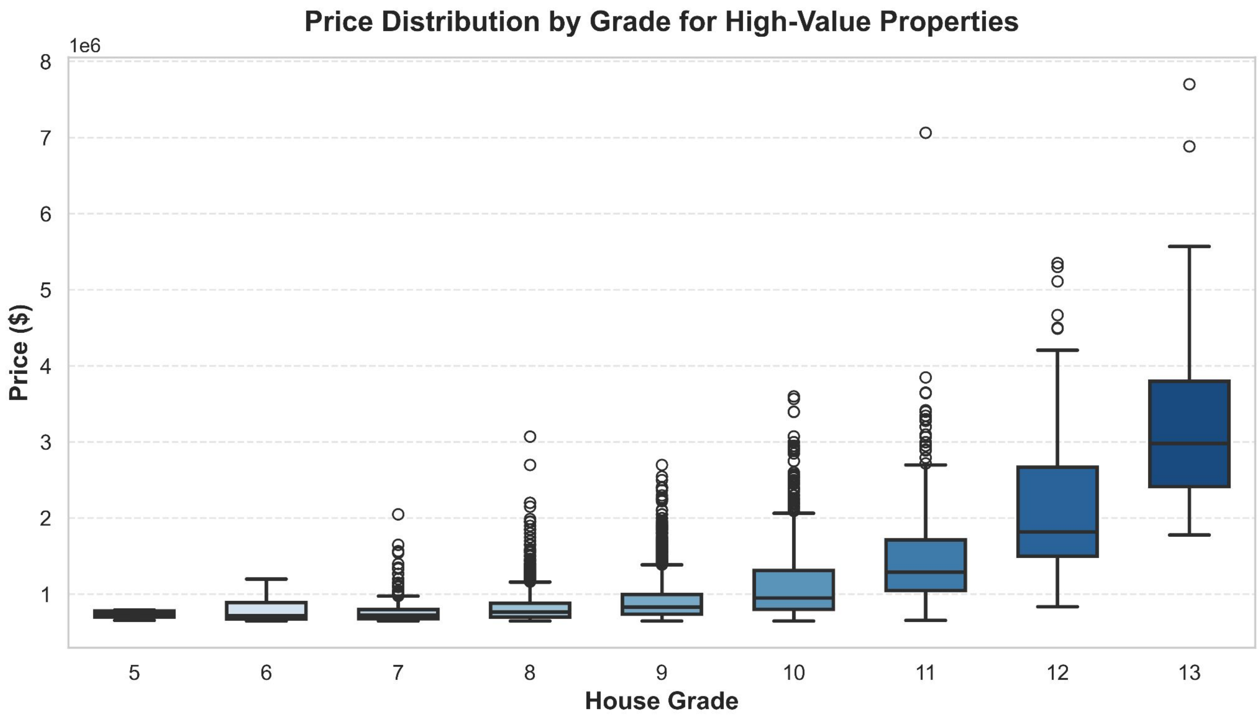


- Room for improvement, but somewhat accurate.

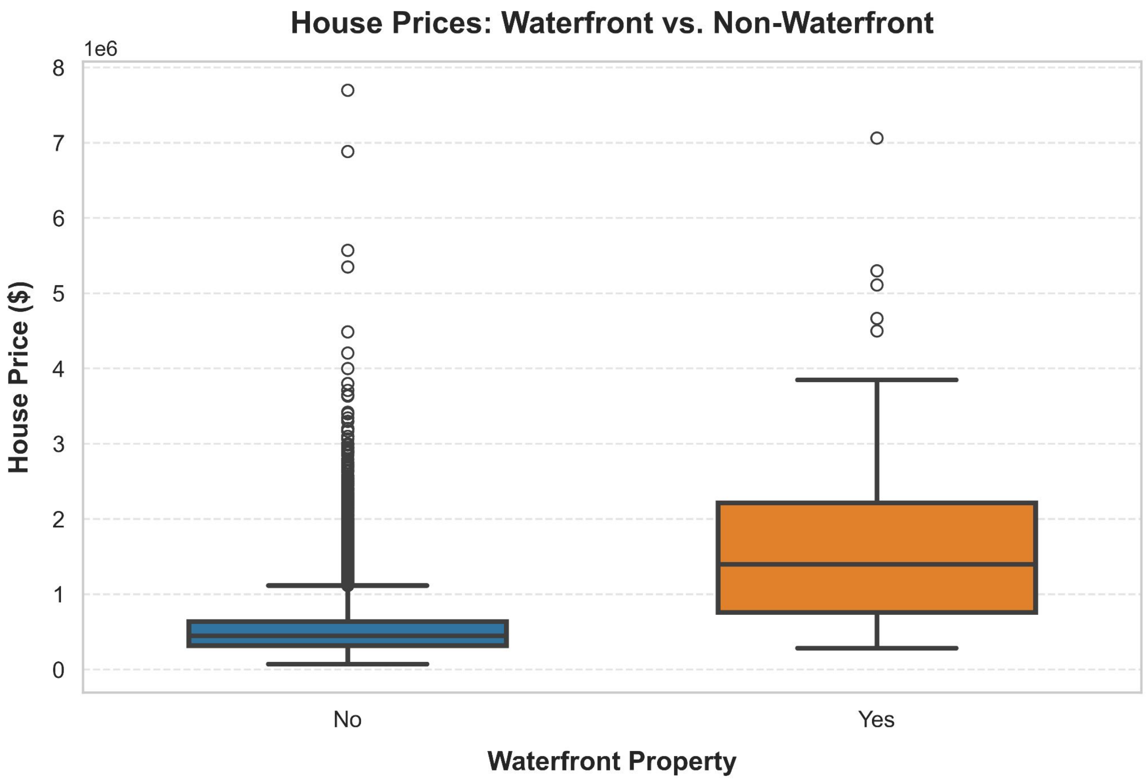
Influential Features by Random Forest Regressor



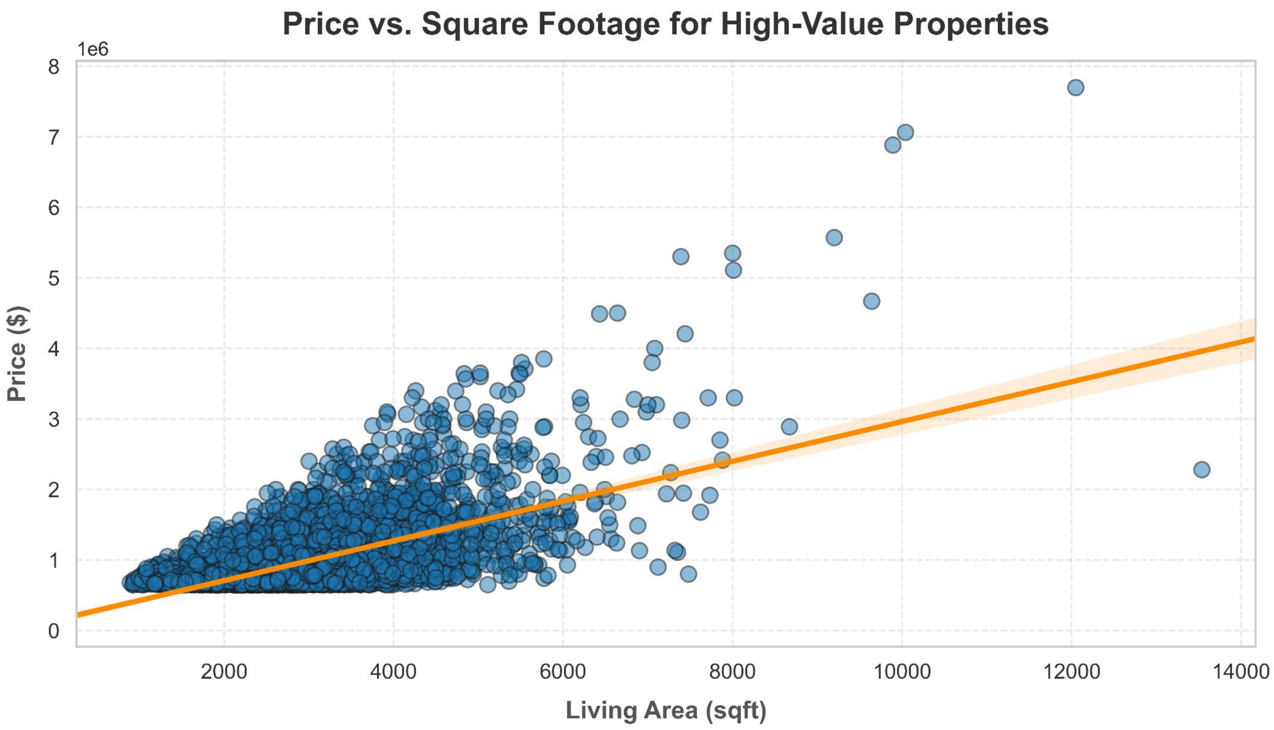
Price by Grade (High-value Properties > $650K)



House Prices w/wo Waterfront



Price vs. Living Space (High-value Properties > $650K)



KNeighborsRegressor

Baseline

	RMSE ($)	R^2
n_neighbours=5 (default)	269117.94	0.498329
n_neighbours=10	**266571.79**	**0.507777**
n_neighbours=20	270203.56	0.494274
n_neighbours=50	276997.80	0.468521

Price > $650k

	RMSE ($)	R^2
n_neighbours=5 (default)	404199.33	0.391227
n_neighbours=10	409527.92	0.375070
n_neighbours=20	418582.92	0.347129
n_neighbours=50	430495.21	0.309441

Benchmark of 7 Models without Price Outliers

	MAE ($)	RMSE ($)	R^2
Linear Regression	87328.13	116316.73	0.678946
Lasso Regression	87328.29	116316.263	0.678949
Ridge Regression	87327.40	116314.75	0.678957
Decision Tree	75147.10	108486.97	0.720715
XGBoost	69213.10	96724.81	0.775342
KNeighborsRegressor	125943.00	16320.77	0.367738
Random Forest	**53559.07**	**77638.15**	**0.856965**

Summary

- **Baseline** and **feature engineered** models are almost similar in **accuracy**
- **Removing multicollinearity** did not improve efficiency
- **Lasso performed better** in all key metrics compared to Ridge
- **KNR** is not a great model for this dataset
- **Random Forest** is the best predictor model in terms of **R^2** and **RMSE**

Thank you for listening!!

Any question?