HomeVision

Predictive Model for Real Estate

The Visionaries

Dylan Jin

Lynn Liu

Andrew Beckerman

Objective

Predict US property value based on property and regional level features using a predictive machine learning model.

- Incorporate key variables to consider when investing in real estate market
- Incorporate “curb appeal” through the use of images

Make housing more equitable for buyers, investors, and sellers

Why are we doing this?

Large Impact

- According to the National Association of Realtors (NAR), the number of homes sold annually has hovered around 6 million units since 2021, with approximately 30% of purchasers being first-time homebuyers.
- The residential real estate market in the United States was valued at USD $36.2 Trillion in 2020, and the commercial real estate market was estimated to be $16 Trillion in 2020.

Existing Solutions

Company Name	Company Stage	Product / Solution Overview	Primary Customer	Key Differentiator
Zillow	Enterprise	**Zestimate** - Predicts value based on property data. The national Zestimate for off-market homes has a median error rate of **7.49%**	Home buyers and real estate agents	Our solution will use images in addition to property data as well as macroeconomic data.
Redfin	Enterprise	Publishes overall price trends	Real estate agents	Our solution will predict pricing on an individual property basis.

Target Audience

HomeVision models the relationship between house features and the price, seeking to serve:

- **Real Estate Investors**: as a tool to educate themselves on important investment features/considerations
- **Home Buyers/Sellers**: as a tool to inform a perspective offer or bid and inform pricing strategy
- **Real Estate Developers**: as a tool to determine potential return upon selling after possible home renovations
- **Real Estate Agents**: as a tool to assist their buyer and seller clients

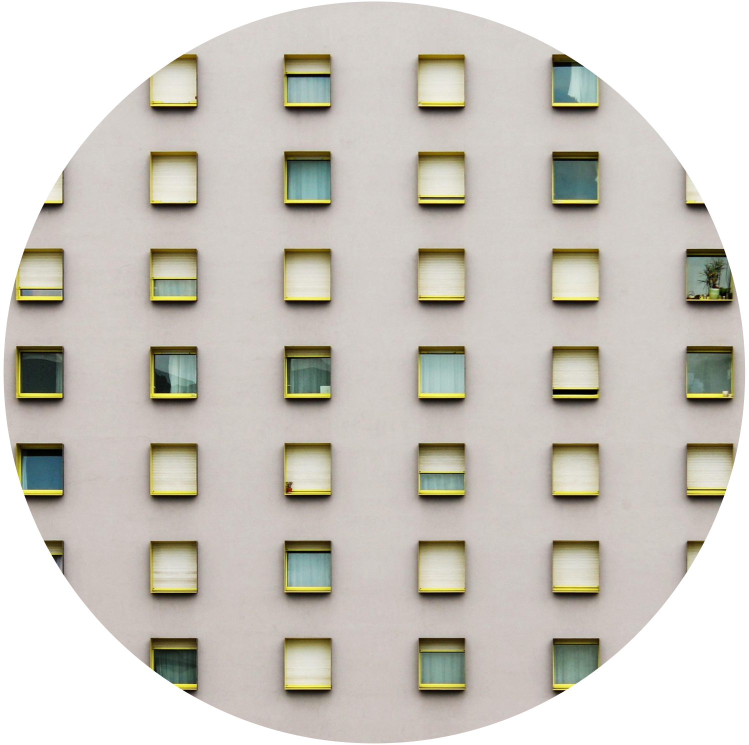
Goals

HomeVision is a residential home pricing tool that considers four inputs (below) to predict the house price:

- **Home Facts**: address, beds, baths, square feet, last sold year/amount, amenities
- **Regional Data:** income, population, GDP, unemployment, average rent, for sale inventory
- **Macroeconomic Indicators:** mortgage rate, SP500 return
- **Curb Appeal:** images of the house facade

User Journey:

1. User lands on the website
2. User uploads an image of a home
3. User keys in basic facts (zipcode, beds, baths, property size)
4. HomeVision returns the predicted home price



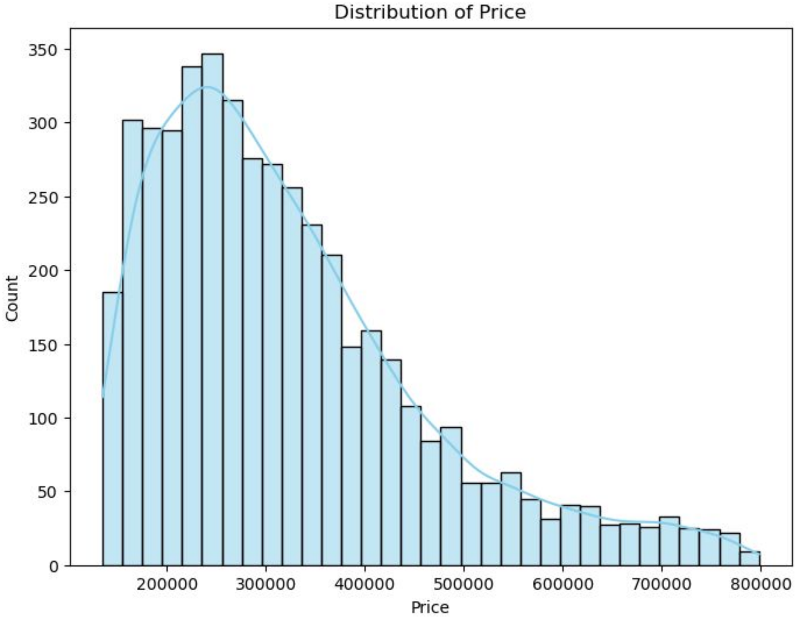
Data Sources

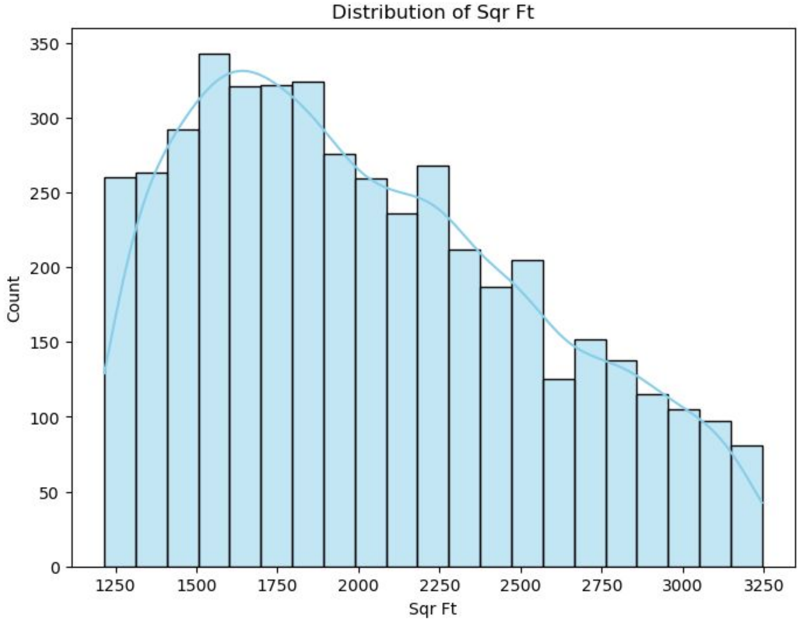
- Primary data source: Trulia property listing dataset (January 2020 and September – October 2019) - available on kaggle.com
  - Currently the largest dataset available with both textual features and images
  - Data are from Trulia, which is reputable
  - Need to clean up/pre-process the data before using
  - Focus on Single Family Home with at least one image
  - Homes have to be from non-auction
- Zillow Data (link)
  - Details on US property characteristics and market trends going back roughly 15 years
- Redfin Data (link)
  - US property characteristics Jan. 2012 - March 2023
- American Community Survey (link)
  - U.S. Census Bureau will be used to incorporate neighborhood-level demographic, social, and economic data

Data Cleaning

- Remove non-feature data. Example: “Uniq Id” is a randomly generated ID for the house. We remove that and will create a simple index for each house
- Convert measurement: convert “lot size” all to sq.ft. as some of them are measured in acres.
- Feature extraction from texts
  - Example feature text: “Single Family Home | $65/sqft | Lot Size: 6,251 sqft | Built in 1938 | 2 Days on Trulia | Floors: Hardwood, Laminate | Parking: Attached Garage | Garage | Stories: 1 | Foundation Type: Concrete | Roof: Shake Shingle | Year Updated: 1975 | MLS/Source ID: 354914”
  - First, extract Single Family homes
  - Second, one hot encoding of categorical variables: such as “Floors”, “Parking”, “Stories”, etc.
  - Extract binary variables: such as “Garage” (=1 if the home has garage)
  - Remove redundant (such as “Build in 1938”) features or non-feature information (such as “MLS/Source ID: 354914”)
  - Checking the feature text for all the houses to make sure all features are included
- Columns with over 25% missing data were discarded
- Removed outliers: Top and bottom 10% of prices
  - Initial Samples: 18,929
  - After Cleaning: 15,187

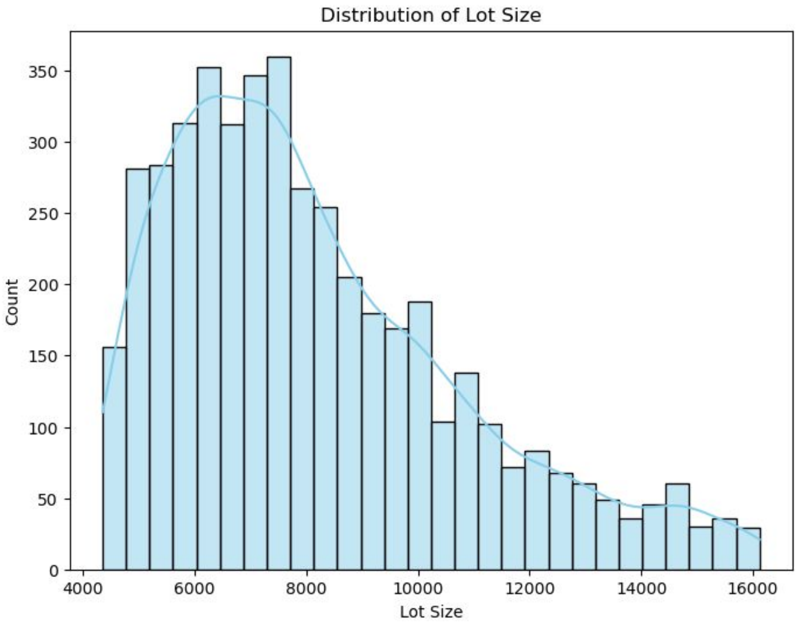
Data Distribution

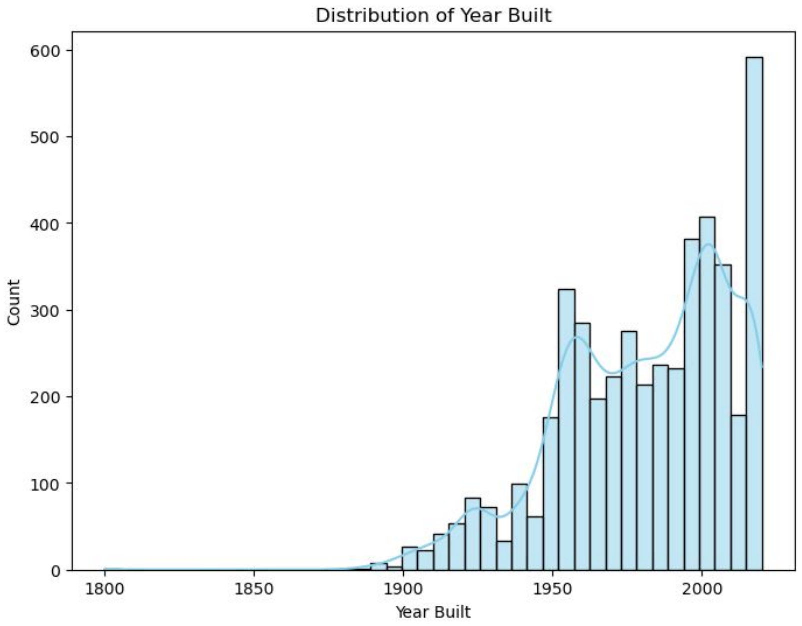




Note: Tails cropped to show detail

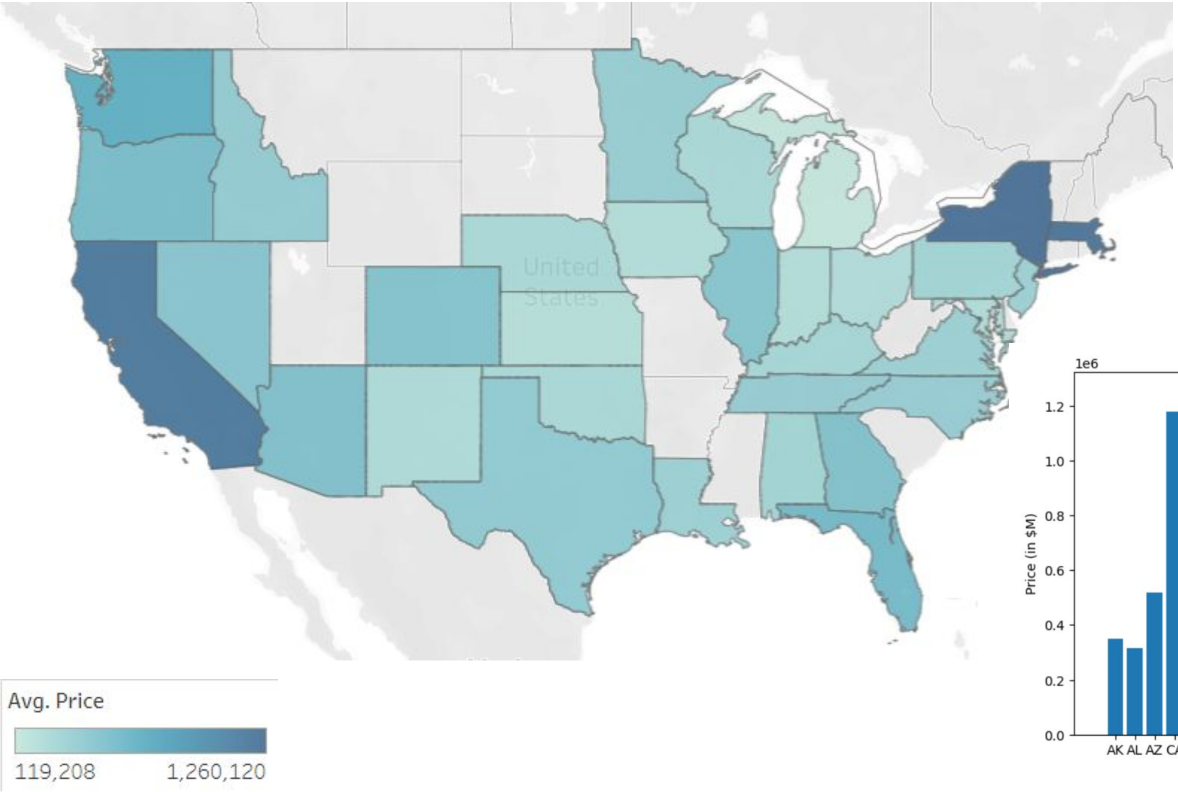
Data Distribution



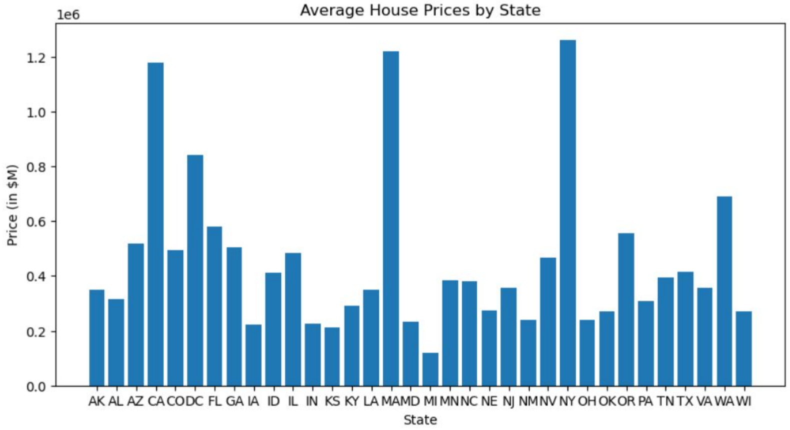


Note: Tails cropped to show detail

Average House Prices by State



- California, New York and Massachusetts have higher average price homes
- Michigan has lower average price home



Modeling

- Standard Modeling
  - Linear and multiple regression are the most widely used models, incorporating various factors like location, square footage, and more to predict property prices

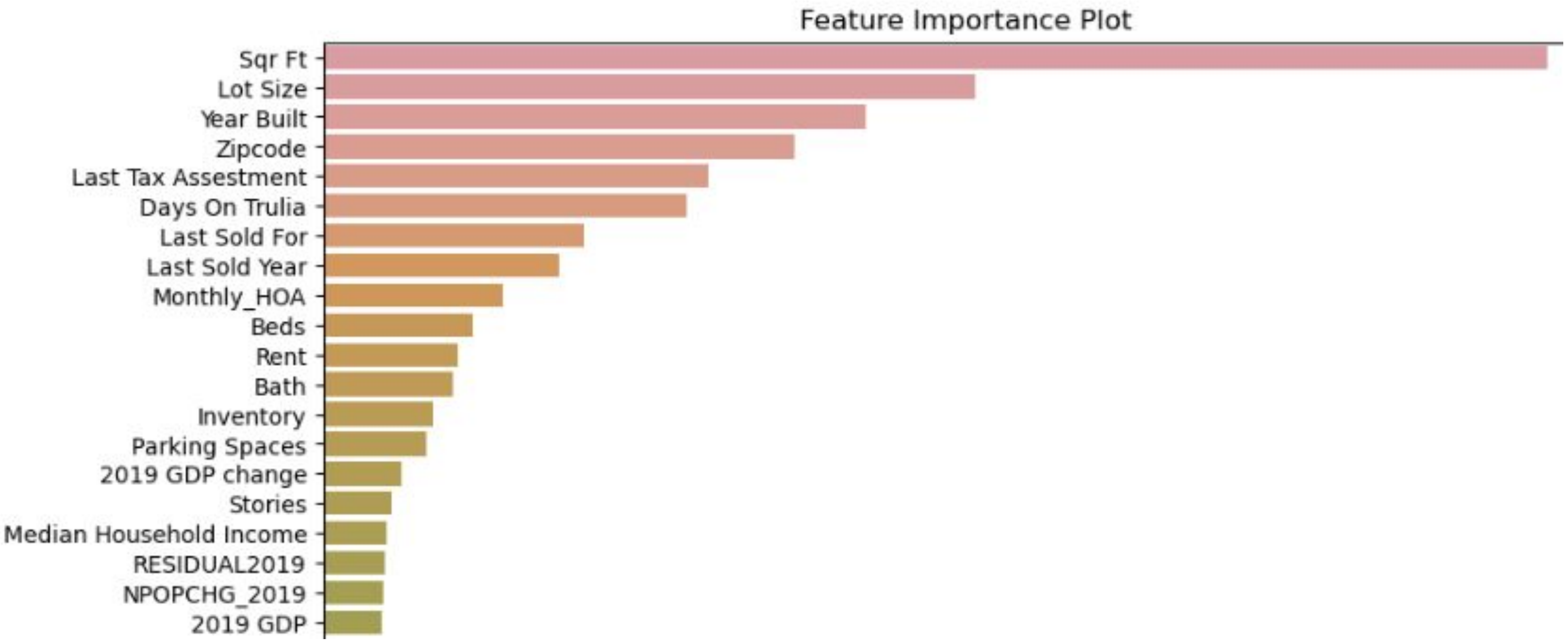
- Image Modeling
  - Currently, the state of the art (SOA) models for Computer Vision task utilize Convolutional Neural Networks (CNNs).
  - Current SOA approaches utilize pretrained models such as VGG16, ResNet, Inception, or EfficientNet

- To combine the models:
  - Transfer Learning:
    - Remove the classification layers from the original pretrained models, keeping the feature extraction layers and weights
    - Pass house images through the feature extraction layers of the pre-trained CNN to obtain a feature vector
  - Incorporate the feature vector along with the other independent variables (location, size, number of bedrooms, etc.) into a regression model

Base Model Performance

Model	Median Percentage Error
Baseline Model (predicts mean of each zipcode)	17.7
Baseline Model (predicts mean overall price)	38.4
Gradient Boosting Model (without regional/macro features)	13.7
Gradient Boosting Model (all features w/ zipcode)	9.8
XGBoost Model (all features w/ location features one hot encoded)	11.1
XGBoost Model (all features w/ location features one hot encoded and lasso)	10.9
XGBoost Model (all features w/ zipcode and lasso)	10.4
XGBoost Model (top 40 features)	10.9
XGBoost Model (top 4 features)	19.6
XGBoost Model (top 40 features and removing outliers)	11.1
Light GBM (all features w/ location features one hot encoded)	11.5
Light GBM (all features w/ all location features not one hot encoded)	11.7
Light GBM (top 40 features)	12.0
Zestimate (Zillow)	8

Top Features



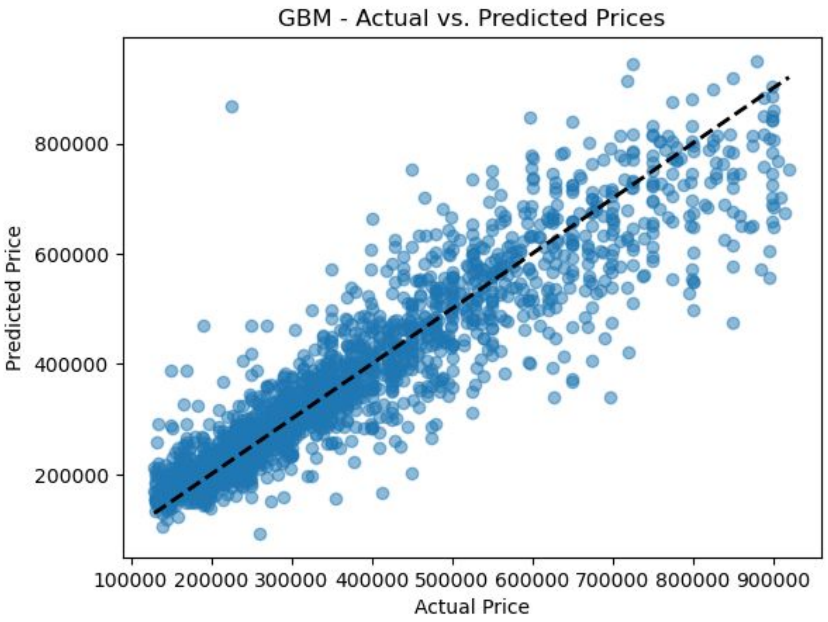
Top Features

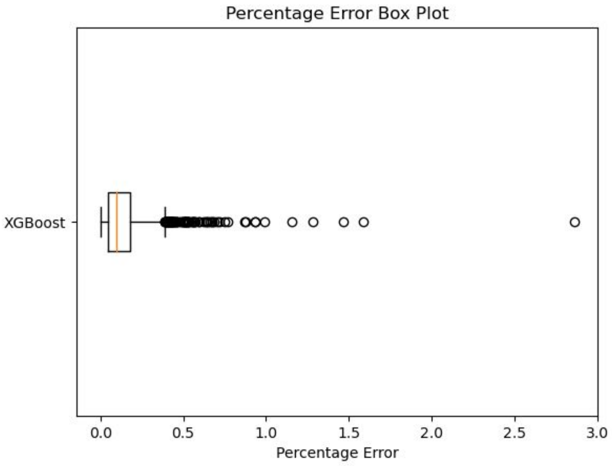
1. Sqr Ft: Square footage of what building occupies
2. Lot Size: Total space of land a building spots on
3. Year Built: Year the property was built
4. Zip Code: Zipcode of property
5. Last Tax Assessment: The last year the property was assessed for taxes
6. Days On Trulia: # of Days on Trulia
7. Last Sold For: The $ the property was last sold for
8. Last Sold Year: The year it was last sold
9. Monthly HOA: $ monthly HOA fee
10. Beds: # of beds in building
11. Rent: $ value of rent
12. Bath: # of bathrooms in building
13. Inventory: Average # of properties in inventory in the country
14. Parking Spaces: # of parking spaces available
15. 2019 GDP change: % 2019 GDP growth rate
16. Stories: # of floors
17. Median Household Income: $ value of 2019 median income per household
18. RESIDUAL 2019: Difference between net change and all sources of change
19. NPOPCHG 2019: Net population change start to end of 2019
20. 2019 GDP: $ value of 2019 GDP

Property Features

Regional/Macro Features

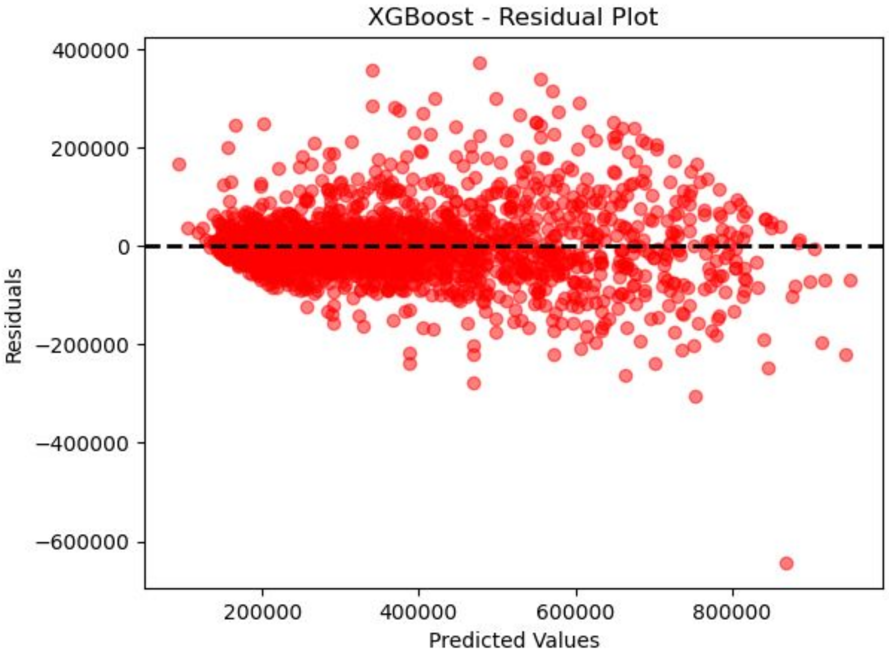
Results - GBM

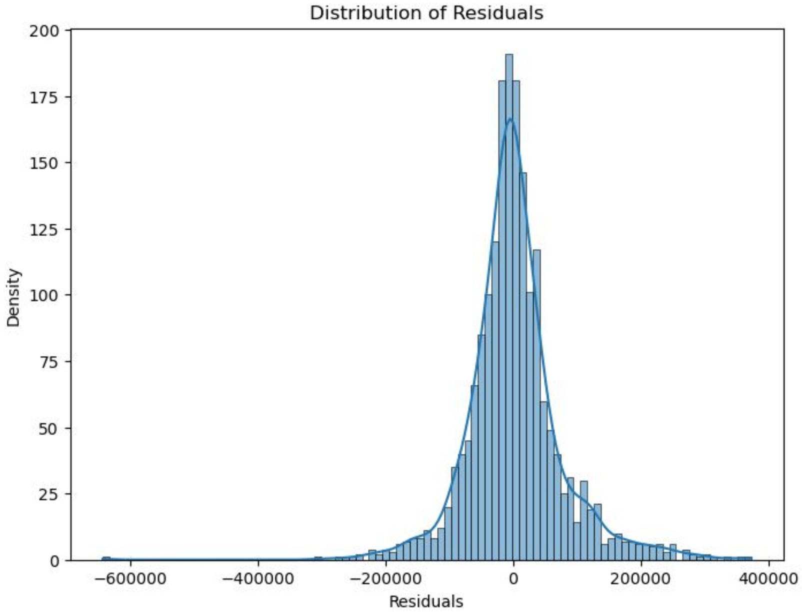




Median Percentage Error: 9.7524

Results - GBM


XGBoost - Residual Plot
400000
200000
0
−200000
−400000
−600000
Residuals
200000
400000
600000
800000
Predicted Values


Distribution of Residuals
200
175
150
125
100
75
50
25
0
Density
−600000
−400000
−200000
0
200000
400000
Residuals

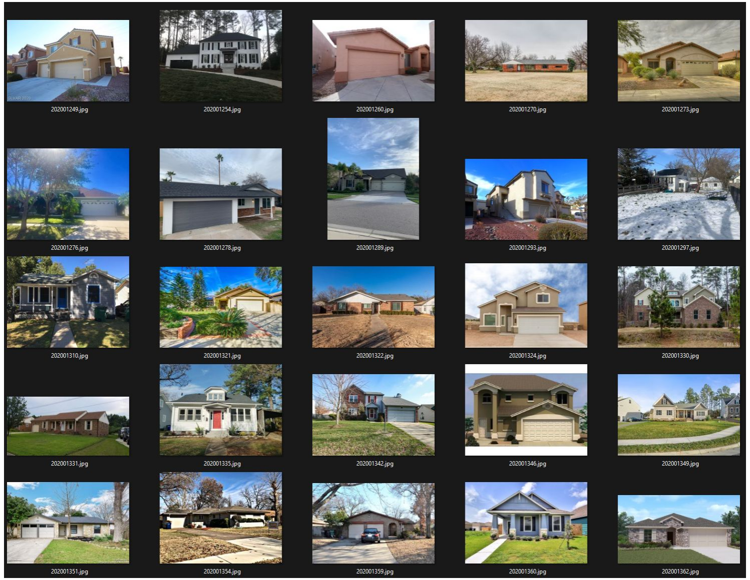
House Image Data

- House Images are provided as a list of URLs in a csv file
    - Created python script to download images for each property into a home folder
- Challenges:
    - Incomplete/Inaccessible data
    - Need to manually review and confirm which properties have viable images.

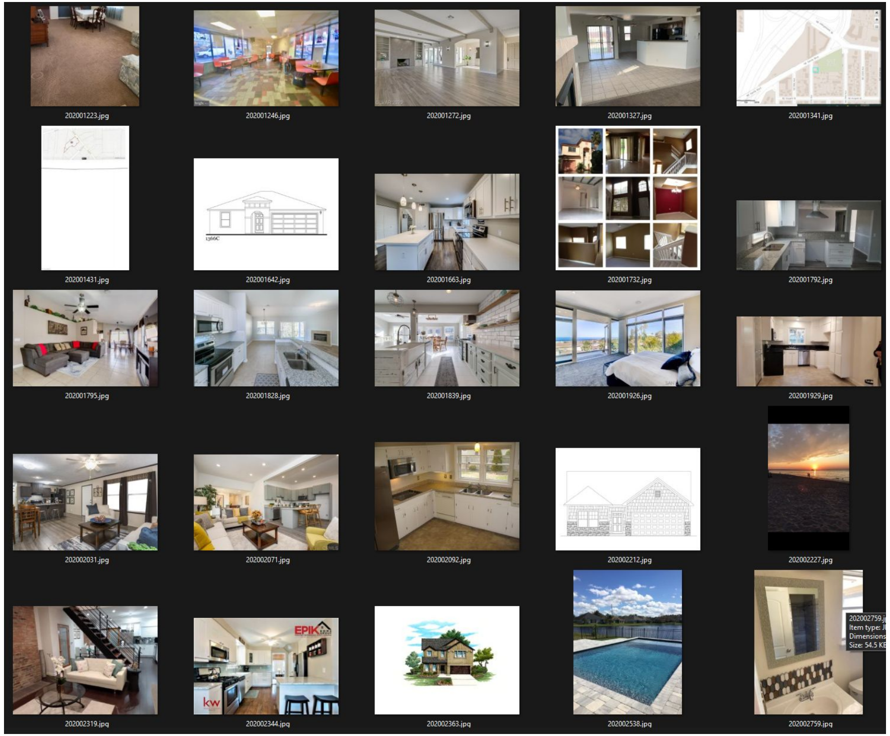
```python
# Function to download an image from a URL and save it with a specified filename
def download_image(url, filename):
    try:
        urllib.request.urlretrieve(url, filename)
        print(f"Downloaded {filename}")
    except Exception as e:
        print(f"Error downloading {filename}: {e}")

# Name of the TSV file
tsv_file = 'single_family_first_image_list_good.txt'

# Directory to save the downloaded images
output_directory = 'image_data'

# Create the output directory if it doesn't exist
os.makedirs(output_directory, exist_ok=True)

# Read the TSV file
with open(tsv_file, 'r') as file:
    reader = csv.DictReader(file, delimiter='\t')
    for row in reader:
        home_id = row['Home_ID']
        # Multi image
        # image_urls = [(column_name, row[column_name]) for column_name in row.keys() if column_name.startswith('Image')]

        # Single image
        image_urls = [(home_id, row[column_name]) for column_name in row.keys() if column_name.startswith('Image')]

        # Create a directory for each home_id
        # Multi image
        # home_directory = os.path.join(output_directory, home_id)
        # os.makedirs(home_directory, exist_ok=True)

        # Single Image
        home_directory = output_directory

        # Download and save each image in the home_directory
        for image_name, image_url in image_urls:
            filename = os.path.join(home_directory, f"{image_name}.jpg")
            download_image(image_url, filename)
```

House Image Data

- **Use primary image**
- **Manual Image Cleaning**
  - 9,200 samples after filtering



“Bad” Images



House Image Data

Photographed Images

Rendered Images

Google Street View Image Data

Good Images

Google Street View Image Data

Bad Images

Satellite Image Data

Images were downloaded using Google's StaticMap API:

- Used Latitude and Longitude
- Zoom Level 20 - highest available
- 18452 samples with Satellite Images vs 9200 Home Images

Image Modeling

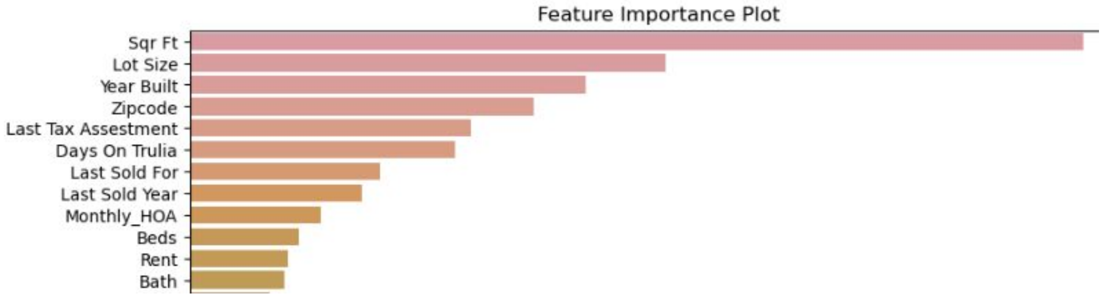
Incorporating Image Data

- Extract Features - EfficientNetV2L
  - Preprocessing to EfficientNet Standards
    - 224 x 224
    - Normalize pixel values
    - < 50 ms / image = 8 minutes
      - 3-5 seconds / image = 13 hours without pre-processing
- Join features to dataframe

```python
### Image Features ###

import tensorflow as tf
from PIL import Image
from tensorflow import keras

# Function to preprocess images to EfficientNet specs
def preprocess_image(image_path):
    img = Image.open(image_path)
    img = img.resize((224, 224))  # Resize image to match EfficientNet input size
    img = np.array(img)  # Convert image to numpy array
    img = img / 255.0  # Normalize pixel values
    img = np.expand_dims(img, axis=0)  # Add batch dimension
    return img

image_folder = "./image_data"
image_features = []

# Iterate over the Home_ID column in your dataframe and extract image features for each corresponding image file
efficientnet = tf.keras.applications.efficientnet_v2.EfficientNetV2L(weights='imagenet', include_top=False)
for home_id in data['Home_ID']:
    image_path = f'./image_data/{home_id}.jpg'
    if os.path.exists(image_path):
        image = preprocess_image(image_path)
        features = efficientnet.predict(image)
        image_features.append(features)
    else:
        image_features.append(None)

# Convert the extracted image features list into a NumPy array
image_features = np.array(image_features)

# Concatenate the image features array with the existing dataframe
df = pd.concat([df, pd.DataFrame(image_features)], axis=1)

columns_to_drop = ['Home_ID']
df = data.drop(columns=columns_to_drop)

### End Image Features ###
```

Image Modeling

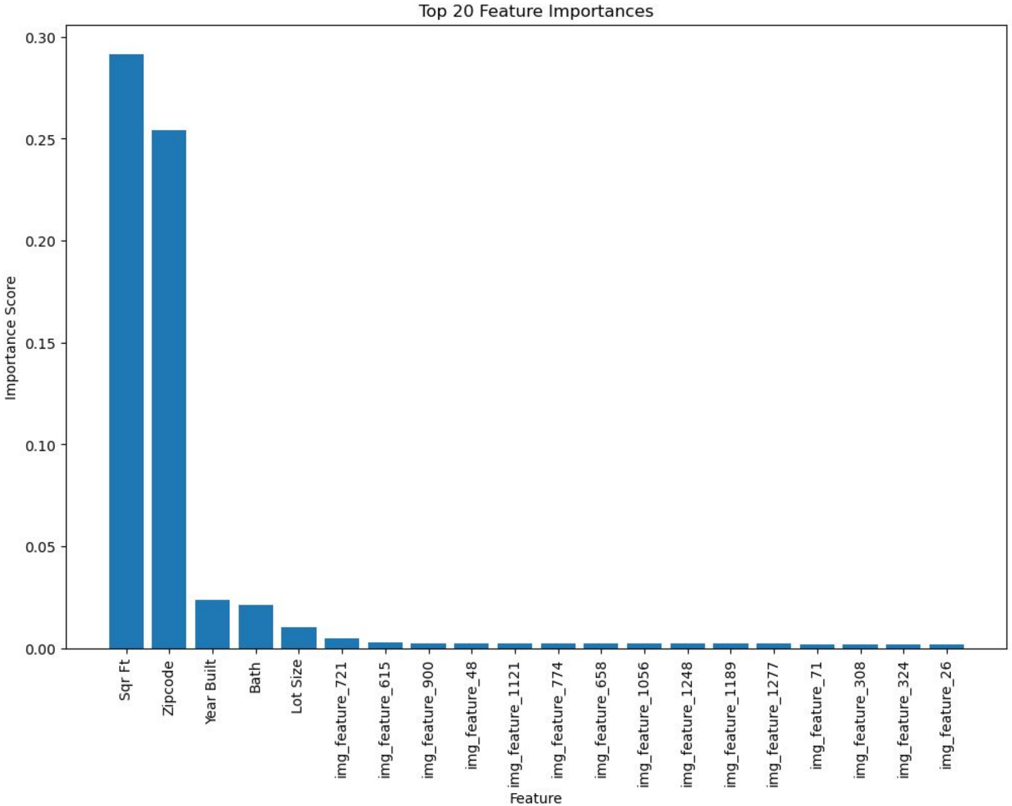
- Gradient Boost Model
- Same cleaned dataset: 9200 Samples
- Reduced features:
  - Sqr Ft
  - Lot Size
  - Year Built
  - Zipcode
  - Beds
  - Bath

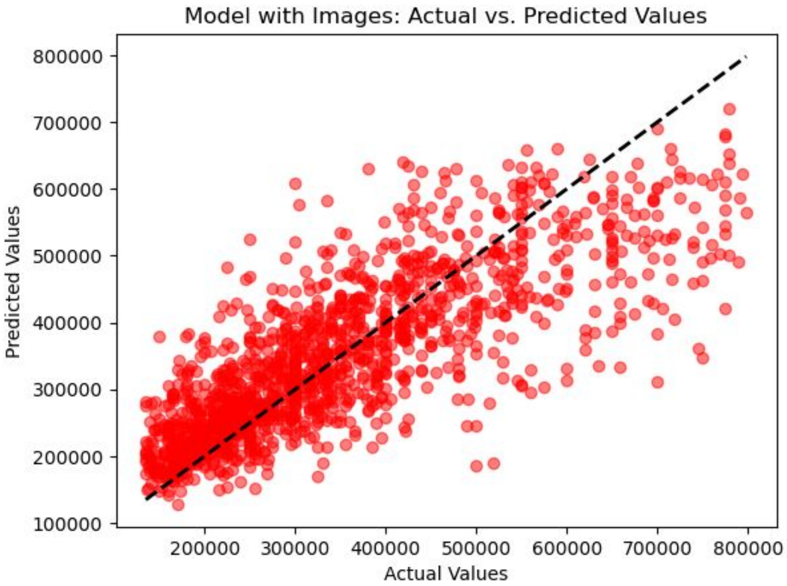


Results - Image Models

Model	Median Percentage Error
Baseline GBM - Reduced Features	17.0
+ All Features	9.1
Home Images - Reduced Features	17.1
+ All Features	12.1
Street View - Reduced Features	15.9
+ All Features	11.9
Satellite - Reduced Features	15.9
+ All Features	12.2

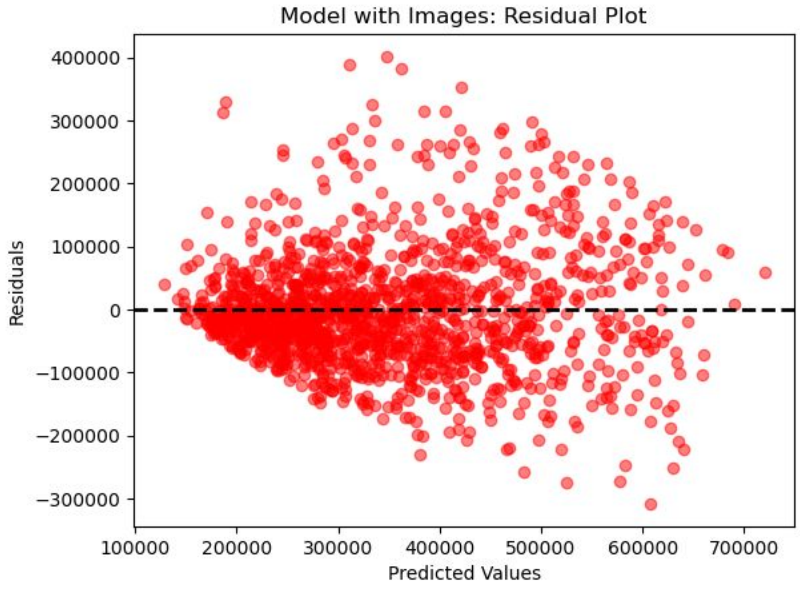
Results - Home Images



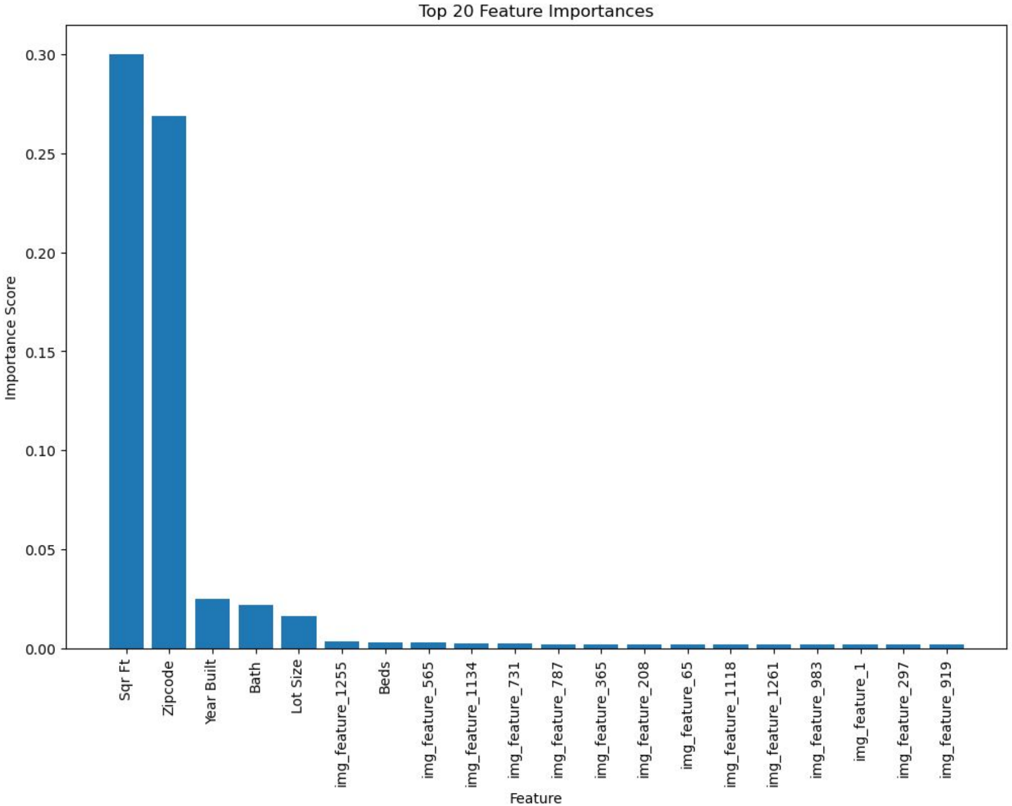


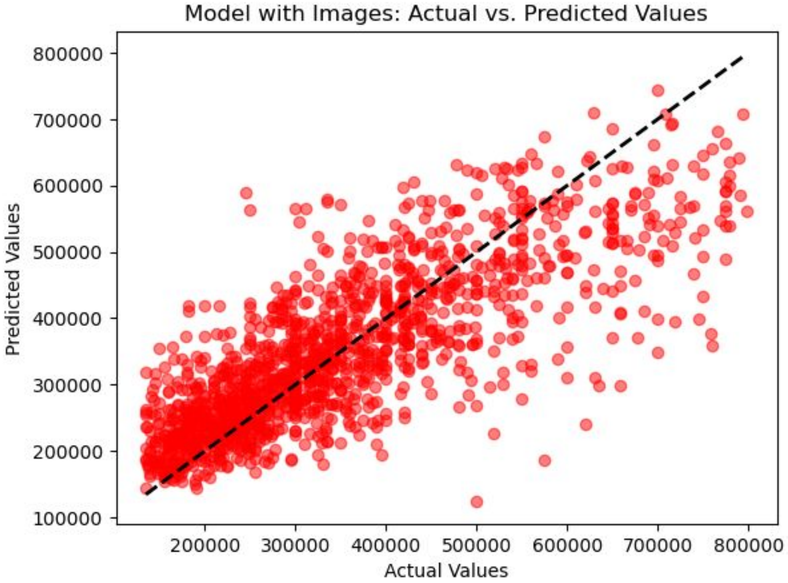
Results - Home Images

Bias towards underestimating price

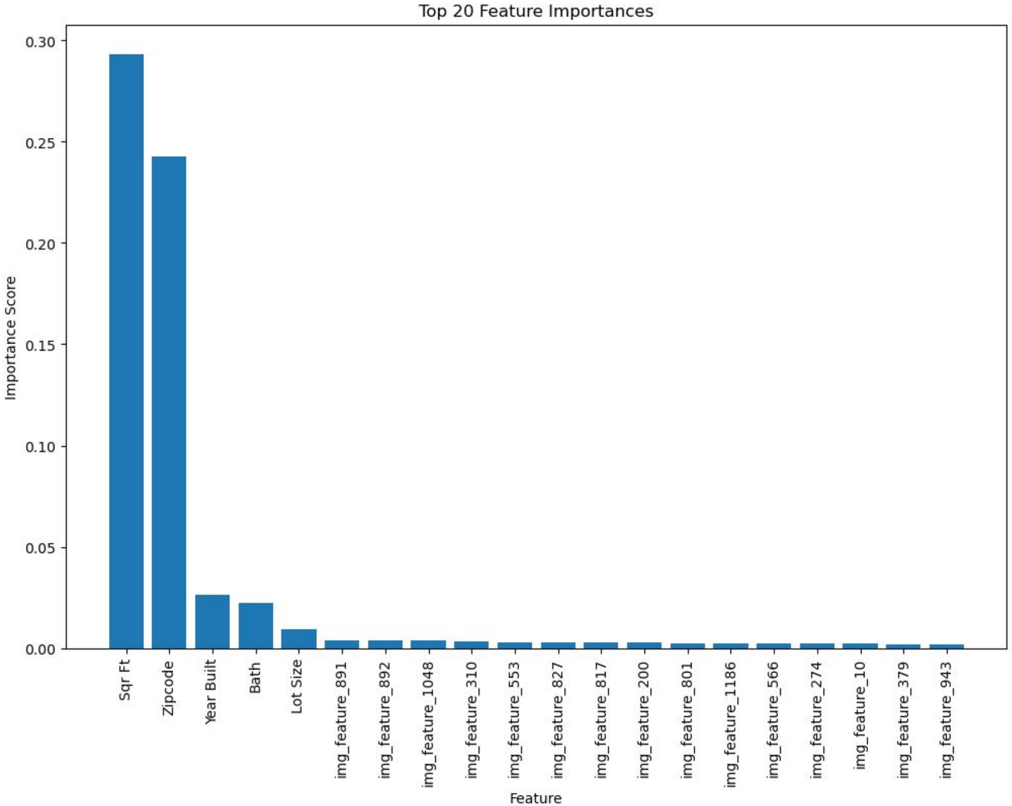


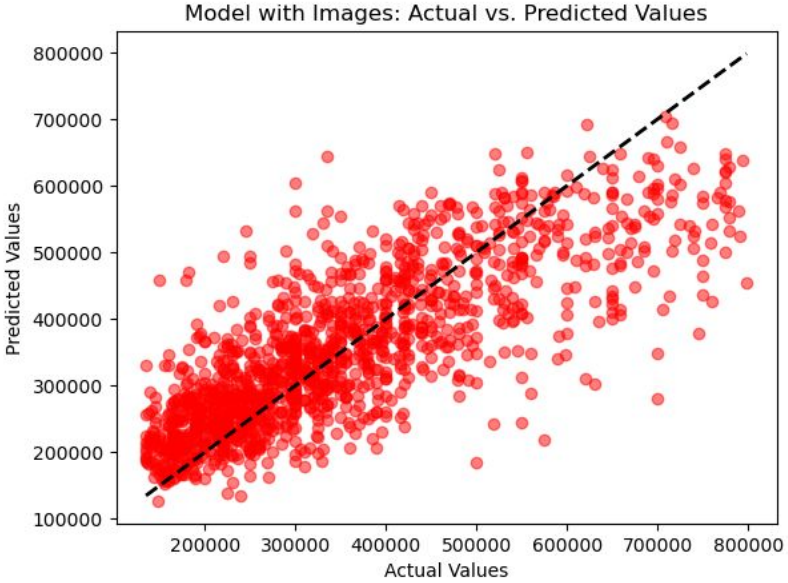
Results - Street View Images





Results - Satellite Images





Conclusions

- Location and property size are the greatest contributing factors to housing price
- Image features were able to improve upon a barebones baseline
- Using Google Street View images were better than either the Home Listing or Satellite images
- Traditional regression models are better at predicting price, but our best imaging model is not far behind (Δ 2-3%)
- Be aware that there is a consistent tendency for the model to underpredict the price.

Thank You

Any Questions?

Appendix

Existing research

- Hedonic models – using textual features such as the number of bedroom, type of house, etc. to predict individual house price (micro-level)
  - Linear regression models (with regularization)
  - Tree-type machine learning models (with boosting)
- Repeated sales/house price index (HPI) – time series problem (macro-level)
  - Traditional, ARIMA models: those are often selected as the baseline model for performance comparison
  - Machine learning models: neural network, LSTM
- Few works on using images to support house price prediction
  - Feature extraction from images (but with a small-sized data)
  - Classification of the level of luxury from images using AMT (Amazon Mechanical Turk)
  - Lacks reliable dataset of houses with both images and sufficiently many textual features

Overview of Datasets

Dataset	# of Columns	# of Rows	Notes
Trulia	73	35K	Many columns have missing data
Combined Dataset with Macro Trends	135	19K	- Only Single Family homes with price/land lots non-empty - At least one image - Non auction homes

Data Cleaning

1. Remove non-feature data. Example: “Uniq Id” is a randomly generated ID for the house. We remove that and will create a simple index for each house
2. Change numeric features, such as price, sq.ft., last tax assessment, to numeric values
3. Convert measurement: convert “lot size” all to sq.ft. as some of them are measured in acres.
4. Feature extraction from texts
   a. Example feature text: “Single Family Home | $65/sqft | Lot Size: 6,251 sqft | Built in 1938 | 2 Days on Trulia | Floors: Hardwood, Laminate | Parking: Attached Garage | Garage | Stories: 1 | Foundation Type: Concrete | Roof: Shake Shingle | Year Updated: 1975 | MLS/Source ID: 354914”
   b. First, extract home type (Single Family, Condo, etc.)
   c. Second, extract numeric/categorical variables: such as “Floors”, “Parking”, “Stories”, etc.
   d. Extract binary variables: such as “Garage” (=1 if the home has garage)
   e. Remove redundant (such as “Build in 1938”) features or non-feature information (such as “MLS/Source ID: 354914”)
   f. Checking the feature text for all the houses to make sure all features are included

Data Processing - Adding regional/macro data

1. Regional data - county level
    a. Income: Median income (Census)
    b. Population: population and net migration (Census)
    c. GDP: GDP level and growth rate (Bureau of Economic Analysis)
    d. Unemployment: labor force and unemployment rate (Bureau of Labor Statistics)
    e. Past 6 months Average rent (Zillow)
    f. Past 3 months Average Inventory data: for major cities only (Zillow)
2. Macro data - time series (2 periods)
    a. Mortgage rate: monthly average (FRED website)
    b. Stock market: SP500 return in the past 6 months (Yahoo Finance)

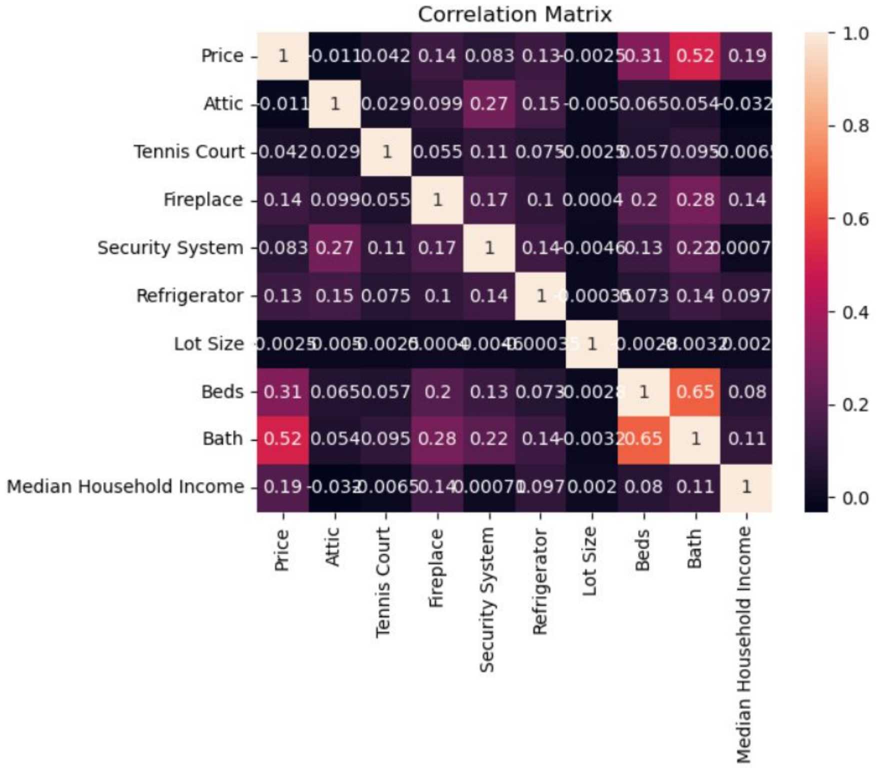
Data Filtering

- Removed listings:
  - No Images
  - No Price
  - Land Lots
  - Auctions: removed all houses under auctions based on description
- Home Types:
  - Apartment
  - Condo
  - Coop
  - Mobile Manufactured
  - Multi Family
  - Others
  - Single Family Home
  - Townhouse

Initial Focus on Single Family Homes:

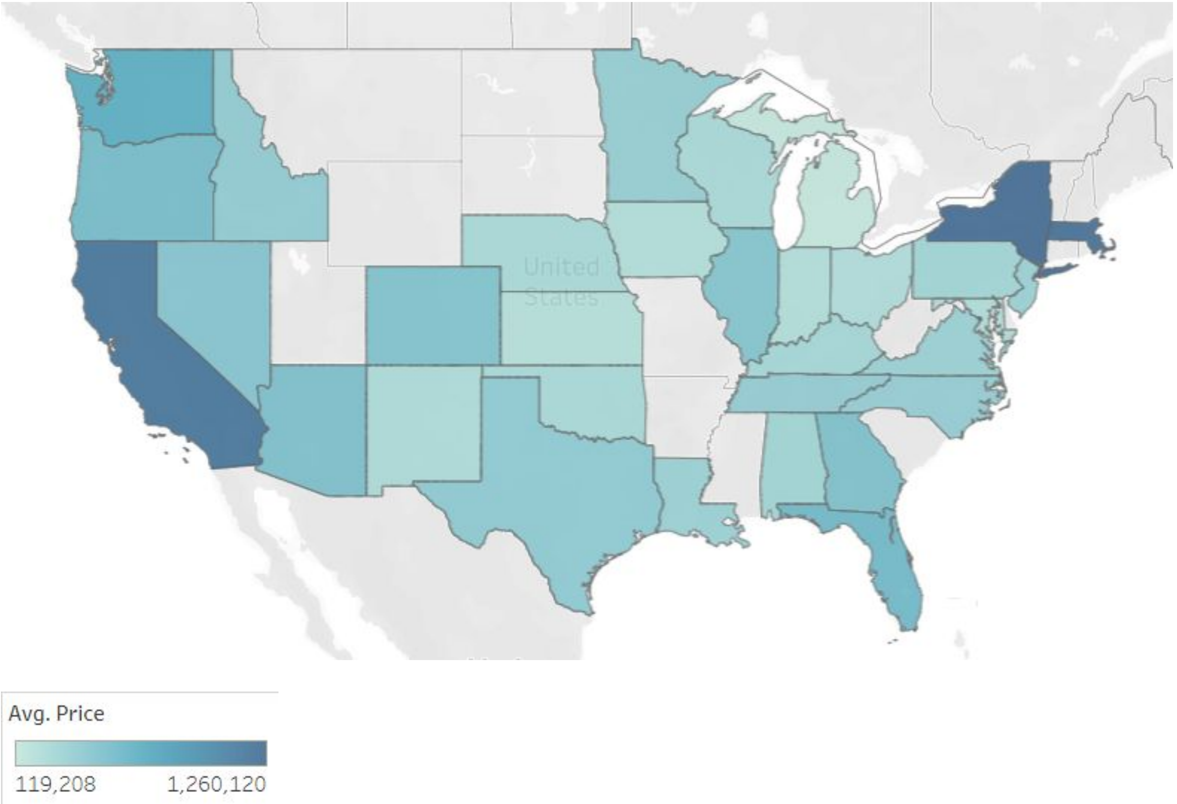
- Other types of homes typically have the same exterior image of the building for multiple units

EDA: Price Correlation with Other Features



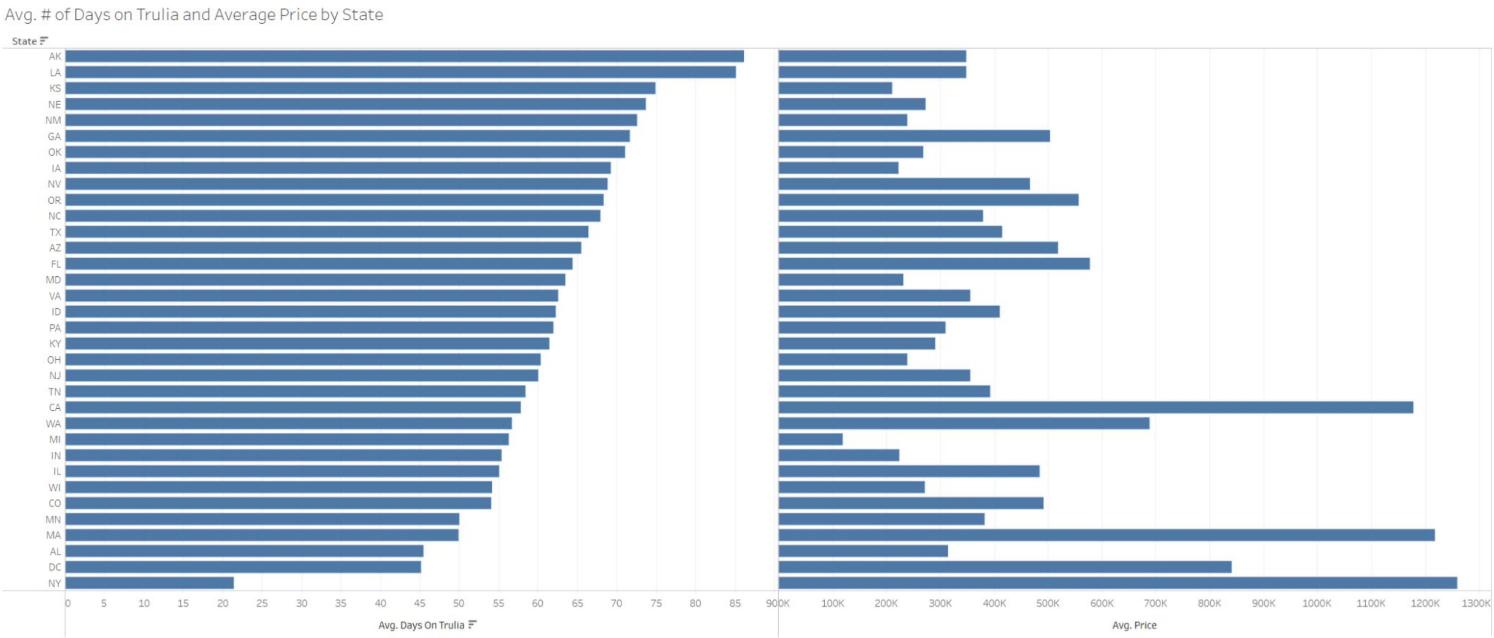
- Price seems somewhat correlated with # of Beds, # of Baths, and Median Household Income

Average House Prices by State



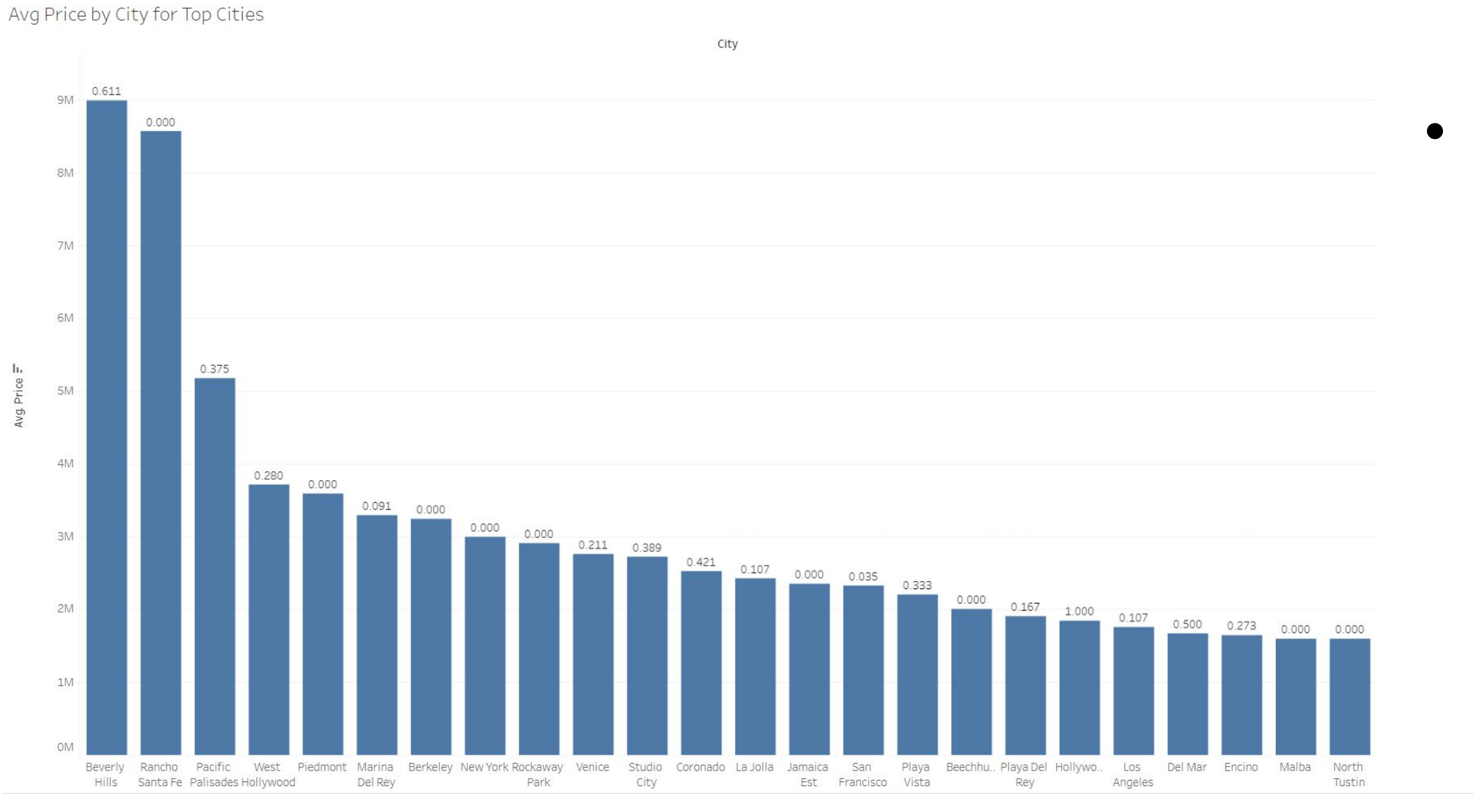
- California, New York and Massachusetts have higher average price homes
- Michigan has lower average price home

Avg # of Days on Market and Avg Price by State



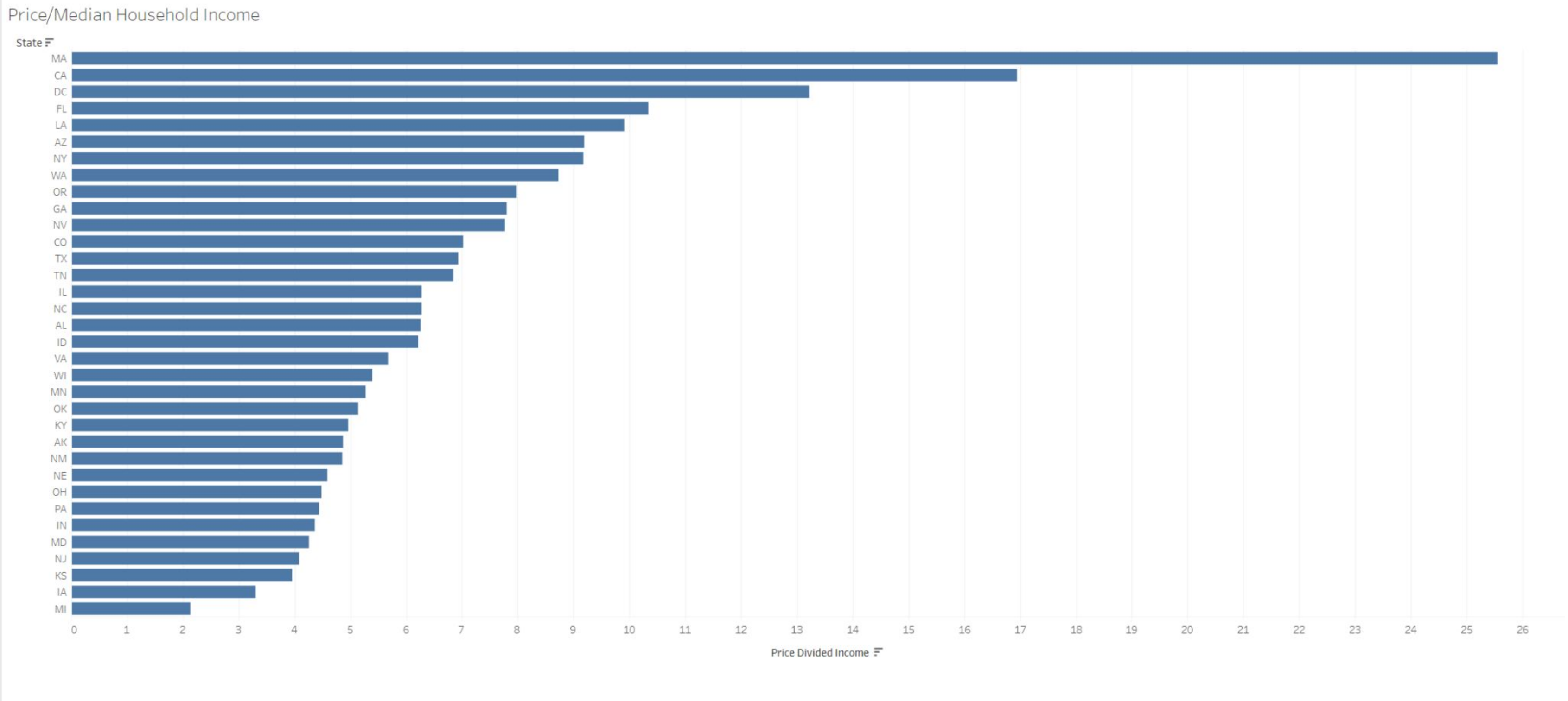
- Seems states with avg higher prices, stay on the market for fewer days

Avg Price by City for Top Cities (and % with Hot Tub/ Spa)

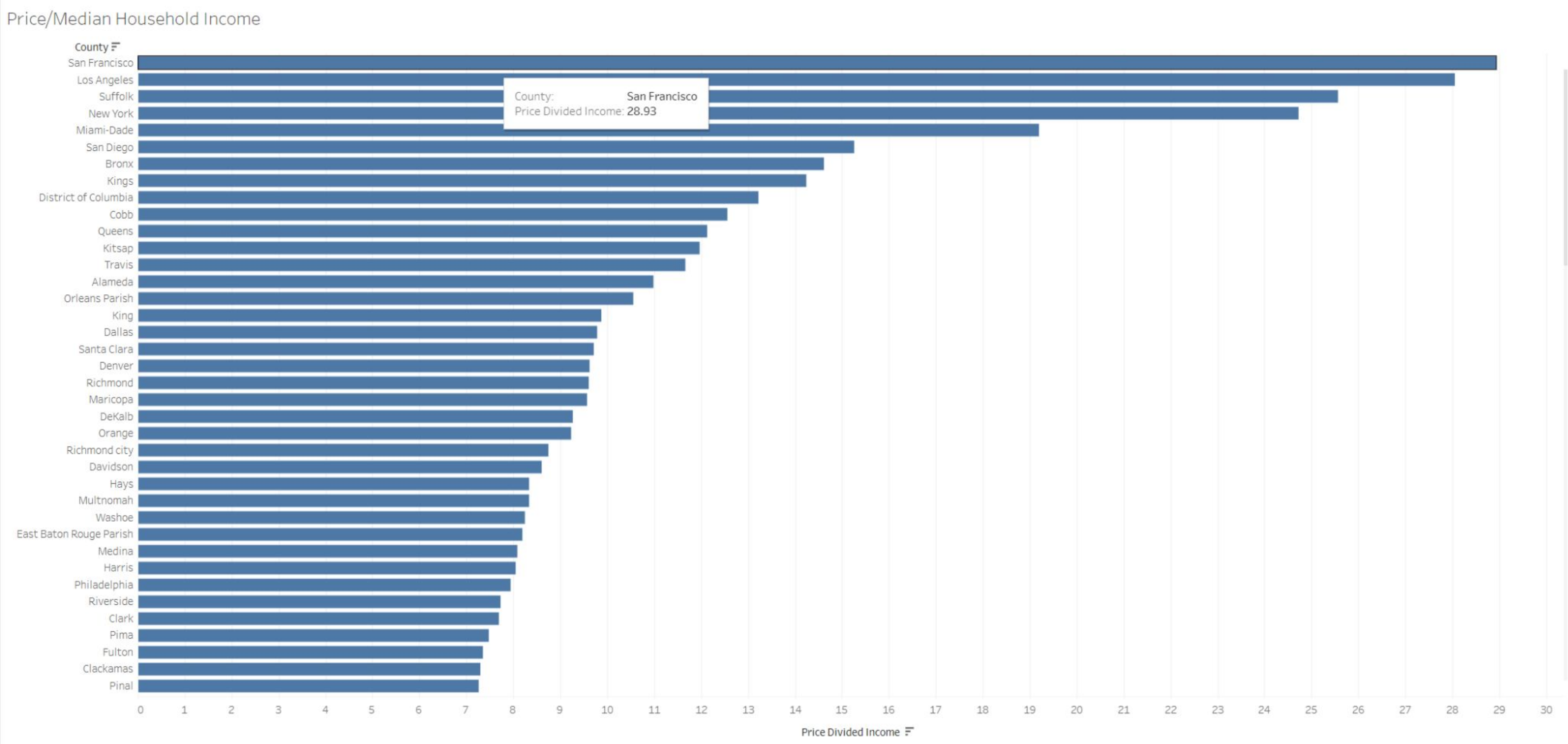


- Many California cities have the highest avg price
    - Beverly Hills is nearly $9M on average
    - San Francisco is $2.3M

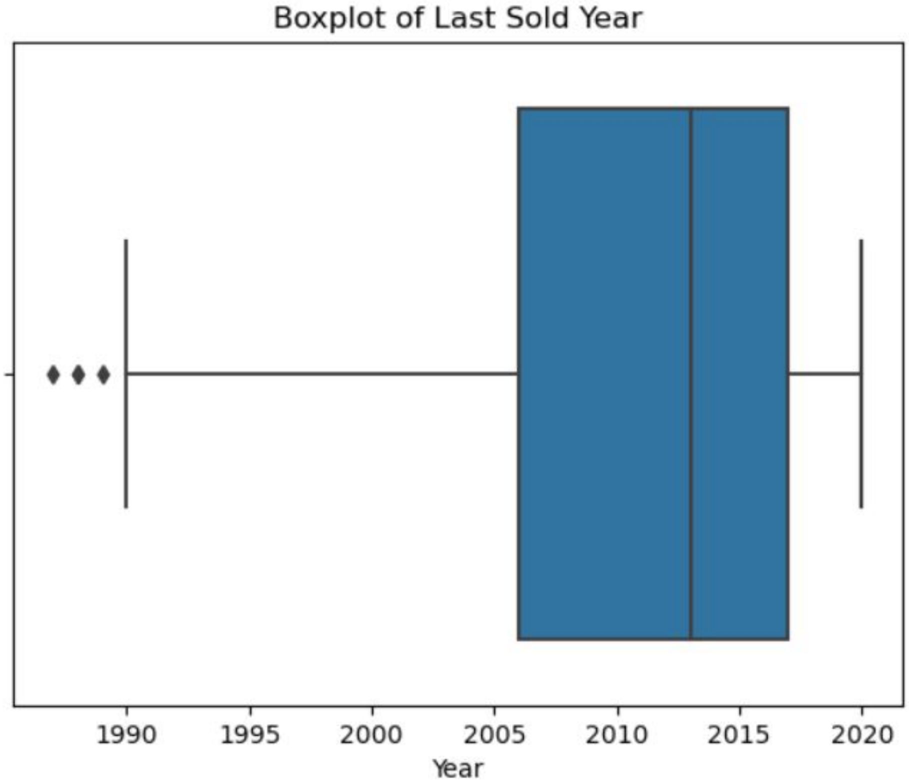
Avg Price of Home / Median Household Income



Price / Median Household Income by County

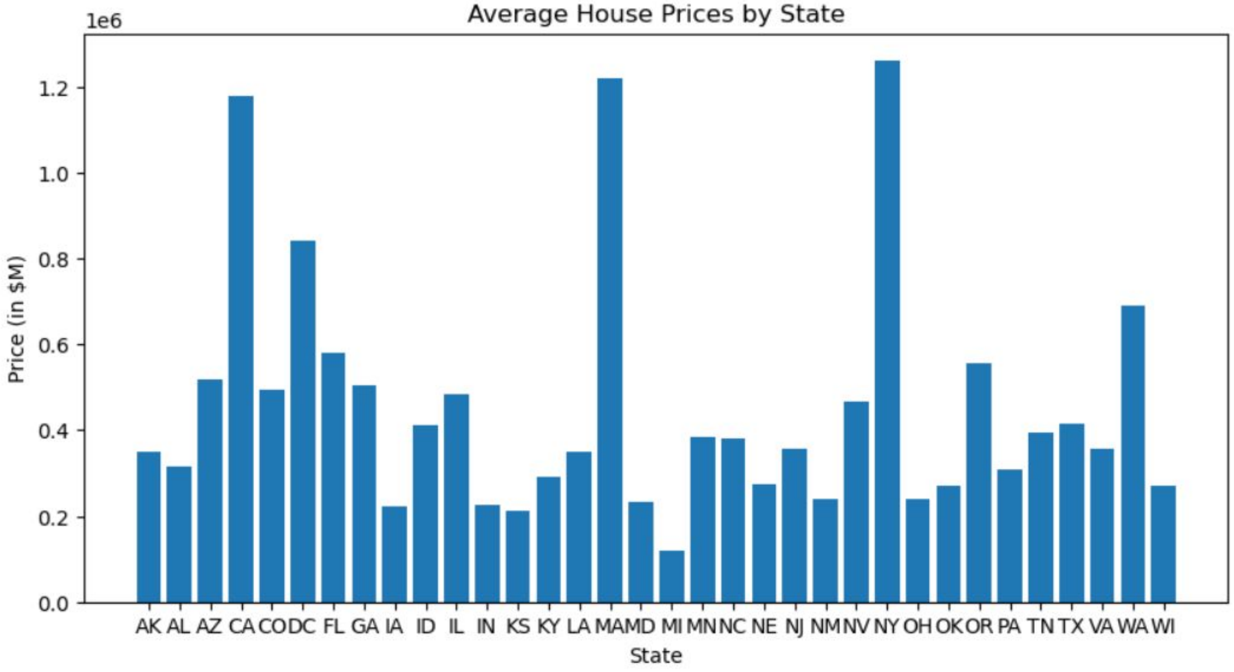


Boxplot Last Sold Year



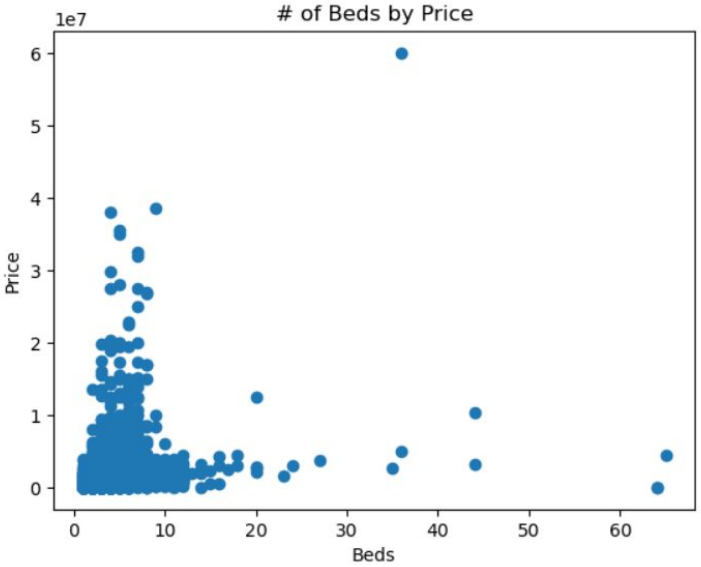
- Over 75% of properties were last sold since 2005

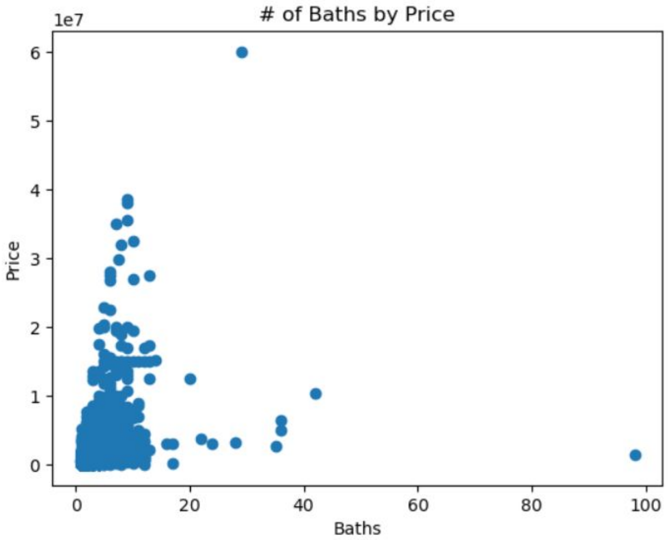
Truila Dataset - House Prices by State



- California, New York and Massachusetts have higher average price homes
- Michigan has lower average price home

Truila Dataset - # of Beds and Baths by Price





- Price of property is slightly correlated with # of Beds and # of Baths

Ethics

The NAR Code of Ethics sets the standard for Realtor business practices.

- 17 articles provide standards for conduct with clients and customers, the public, and other Realtors.
- It's their duty to protect their client's best interest, but treat all parties involved in a transaction honestly.

Data Collection - Low Risk

- Home image, home facts (beds/baths/square feet/etc)
- No personally identifying data is collected
- Do not plan to store collected user data
- Users unlikely to be minors

Concerns

- Which groups are overrepresented or underrepresented in your datasets
    - Will this product may work better for some people over others?
        - Redlining
- Price metrics (current list, selling, appraised value, current estimated value, assessed value, etc.)
    - Which is the “better” way to be wrong (over/underestimate)?
    - Balance needs of the different users
    - Price / Sq ft may be less noisy