Property Valuation using Satellite Imagery Analysis

The objective of this project is to create a Multimodal Regression Model capable of predicting property prices based on structured Tabular-data(numerical/categorical) and unstructured Image-data. Traditional methods using Tabular data use quantative factors which are intrinsic to the house like area, grade, number of rooms etc., satellite images take into consideration extrinsic factors like green cover, curbs, road-density, neighbourhood layout that might affect the house prices.

Hypothesis

We aim to see if factors such as curb appeal and neighbourhood characteritics such as green cover, road density etc. influence prices in an area. Since this data can not be tabulated, we are using satellite imagery to extract meaningful features which might affect house prices

Approach

Rather than treating both modalities equally from the start, we adopted a residual learning strategy. First, a strong tabular model (XGBoost) is trained to capture the majority of variance in house prices. This model serves as a baseline predictor. Next, a convolutional neural network (EfficientNet-B0) is trained on satellite images to predict the remaining error (residual) made by the tabular model. Finally, the final price estimate is obtained by adding the CNN-predicted correction to the baseline output.

This design is motivated by two key insights:

Tabular features already explain most of the price variance. Directly asking the CNN to predict price often leads to underperformance or overfitting due to limited image-specific signal.

Satellite imagery provides weak but complementary cues. Environmental factors such as greenery, proximity to water, road structure, and neighborhood density are not fully encoded in structured attributes, but can help correct systematic errors in the tabular model.

Overall, our approach treats satellite imagery not as a replacement for structured data, but as a contextual refinement layer that improves predictions where tabular models alone are insufficient.

Model Architecture

Tabular Baseline Model (XGBoost)

We use an XGBoost regressor trained on numerical housing attributes. To stabilize variance and improve learning, the target variable (price) is log-transformed.

Role:

Captures dominant price drivers such as square footage, grade, location, and number of bathrooms.

Provides a strong baseline prediction y_tab.

Residual CNN (EfficientNet-B0)

A convolutional neural network based on EfficientNet-B0 (ImageNet-pretrained) processes satellite images centered on each property. Instead of predicting price directly, the CNN is trained to estimate the residual error:

```
                         r = y_true - y_tab
```

The CNN learns visual patterns associated with systematic under- or over-valuation by the tabular model, such as:

Surrounding greenery or water bodies,

Density and structure of nearby buildings,

Road networks and open spaces.

Regularization techniques such as freezing most backbone layers, data augmentation, and dropout are used to prevent overfitting.

To get the final prediction:

```
         y_final = y_tab + α.r
```

where:

y_tab is the tabular prediction,

r is the CNN-predicted residual,

α is a scaling factor (used conservatively to avoid over-correction).

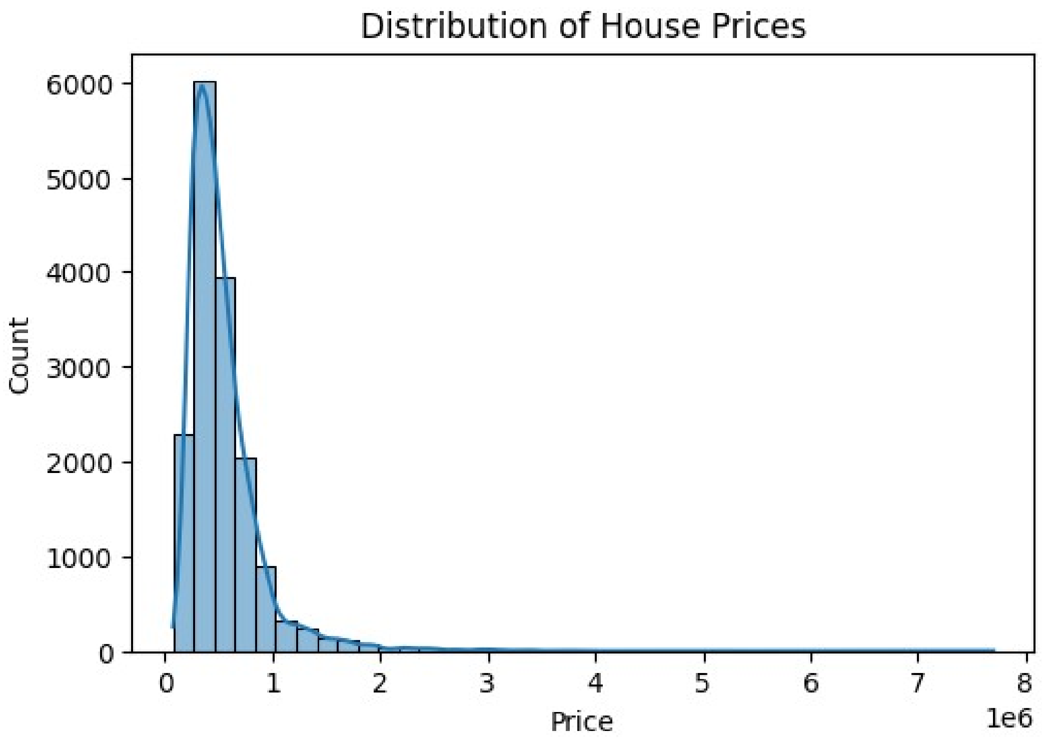
If a satellite image is unavailable, the model automatically falls back to:

```
         y_final = y_tab
```

This design ensures both robustness and interpretability, with the CNN acting as a corrective layer rather than an opaque replacement. This is due to the already strong baseline using tabular data.

EDA

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

data_path = "/kaggle/input/train-data/train(1).xlsx"
property_data = pd.read_excel(data_path)

plt.figure(figsize=(6,4))
sns.histplot(property_data["price"], bins=40, kde=True)
plt.title("Distribution of House Prices")
plt.xlabel("Price")
plt.ylabel("Count")
plt.show()
```



The house prices are left-skewed, and majority of houses lie within a very small range.

```
property_data["date"]=pd.to_datetime(property_data["date"])
corr = property_data.corr()["price"].sort_values(ascending=False)
print(corr)
```

```
price            1.000000
sqft_living      0.700933
grade            0.664266
sqft_above       0.602648
sqft_living15    0.581781
bathrooms        0.525487
view             0.390534
sqft_basement    0.320301
```

```
lat              0.310008
bedrooms         0.304454
floors           0.251428
waterfront       0.245221
yr_renovated     0.133075
sqft_lot         0.088526
sqft_lot15       0.076060
yr_built         0.048307
condition        0.031333
long             0.024279
date             0.004310
id              -0.020260
zipcode         -0.054517
Name: price, dtype: float64
```

Correlation Analysis: Factors Affecting House Prices

This analysis aims to understand the extent to which different features influence house prices. Since **correlation measures only linear relationships**, some features may have their true importance **under-represented** if their effect on price is non-linear.

Overall, the results indicate that **property size, construction quality, and location-based features** play the most significant role in determining property value. Attributes such as *view* and *neighborhood characteristics* highlight the importance of spatial and environmental context.

High Correlation Features

1. `sqft_living`

- Shows the **strongest correlation** with house prices.
- Larger living areas generally command higher prices due to increased usable space and land value.

2. `grade`

- Indicates the **overall quality of construction and architectural design**.
- Higher-grade houses typically use better materials and meet superior construction standards, leading to higher costs and market value.

3. `sqft_above`

- Represents the living area **above ground level**.
- Since it constitutes a major portion of `sqft_living`, it naturally ranks high in correlation with price.

4. `sqft_living15`

- Captures the **average living area of the 15 nearest neighboring houses**.

- This reflects the influence of **neighborhood characteristics** on valuation and suggests that houses of similar sizes are often clustered together.

5. `bathrooms`

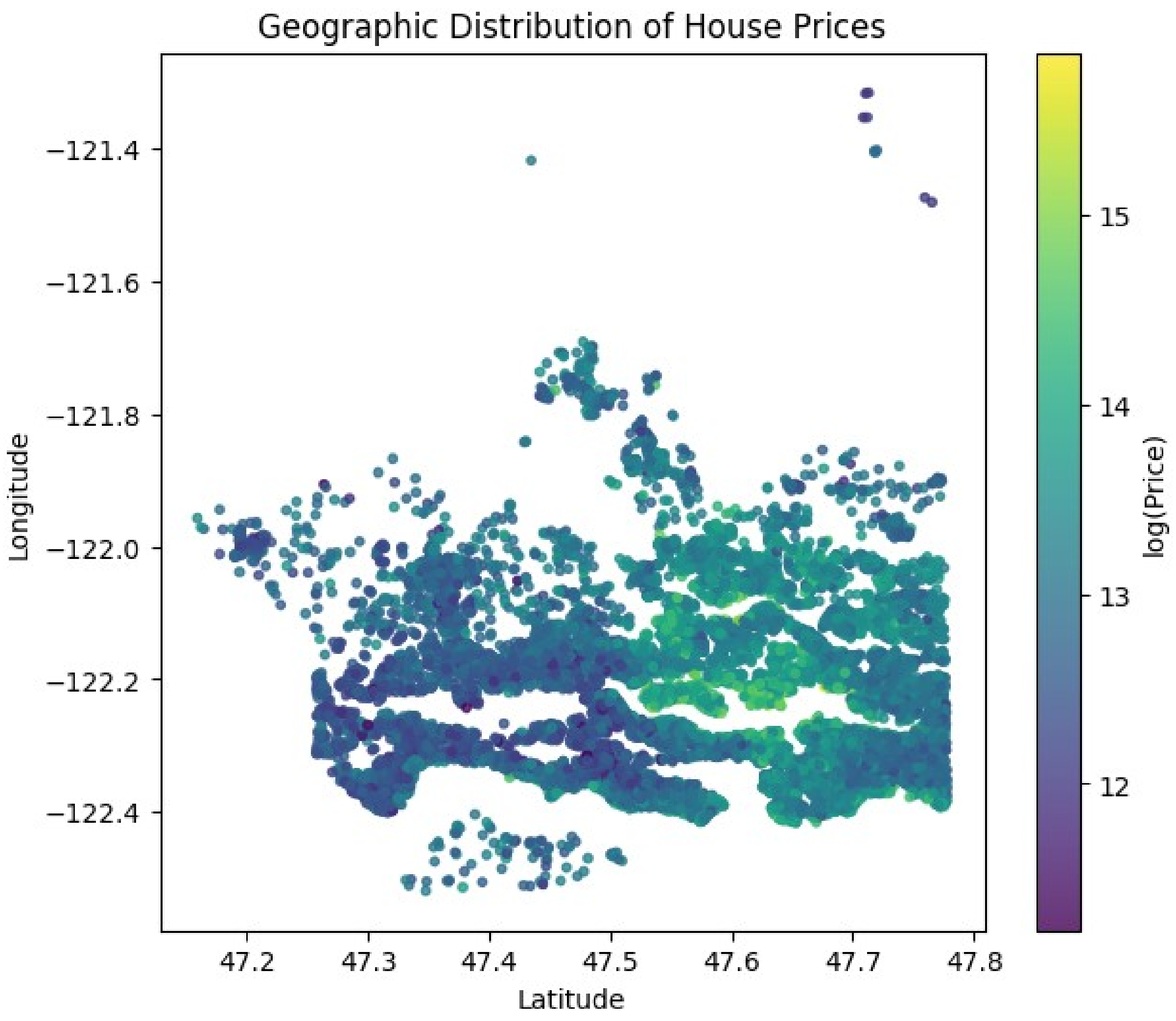
- A higher number of bathrooms generally corresponds to a **larger and more premium property**, contributing to increased price.

6. `view`

- Properties with **better scenic views** tend to be priced higher.

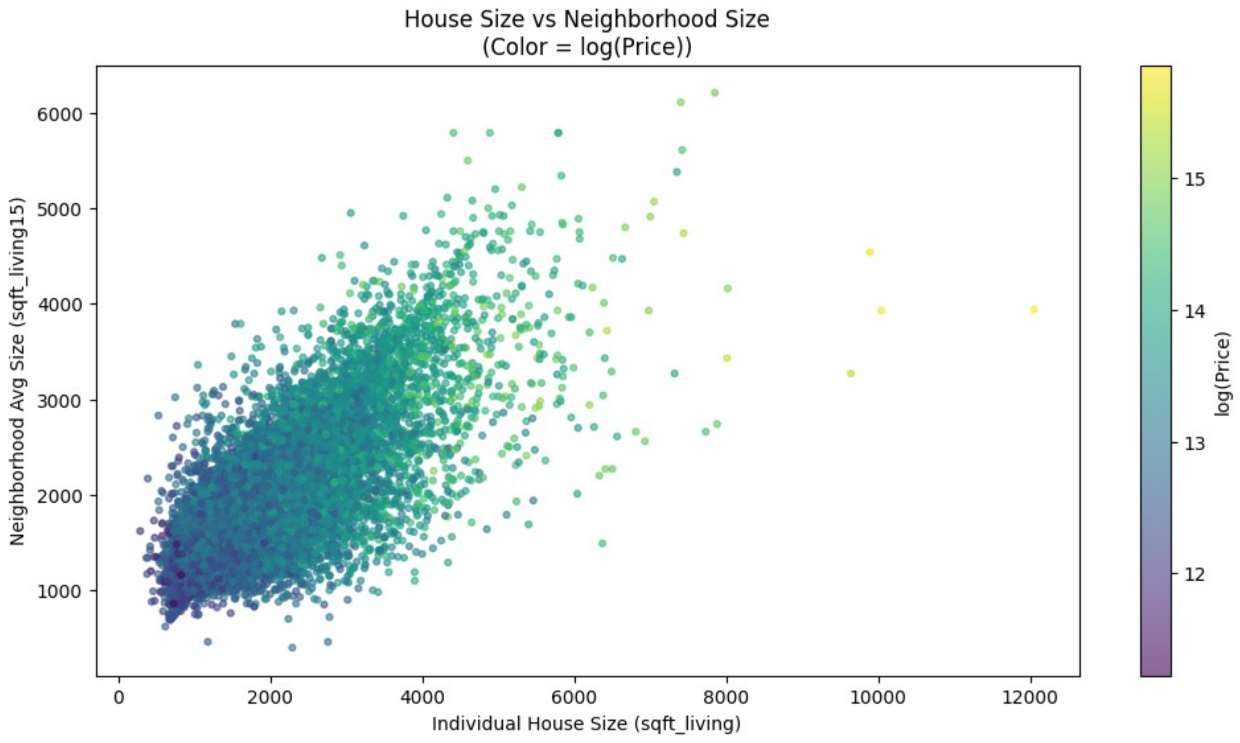
- This highlights the importance of **visual and environmental factors**, showing the need for **satellite imagery** to capture contextual information that traditional tabular data may miss.

```python
plt.figure(figsize=(7,6))

plot = plt.scatter(
    property_data['lat'],
    property_data['long'],
    c=np.log1p(property_data['price']),
    cmap='viridis',
    s=10,
    alpha=0.8
)

plt.colorbar(plot, label="log(Price)")
plt.title("Geographic Distribution of House Prices")
plt.xlabel("Latitude")
plt.ylabel("Longitude")
plt.show()
```

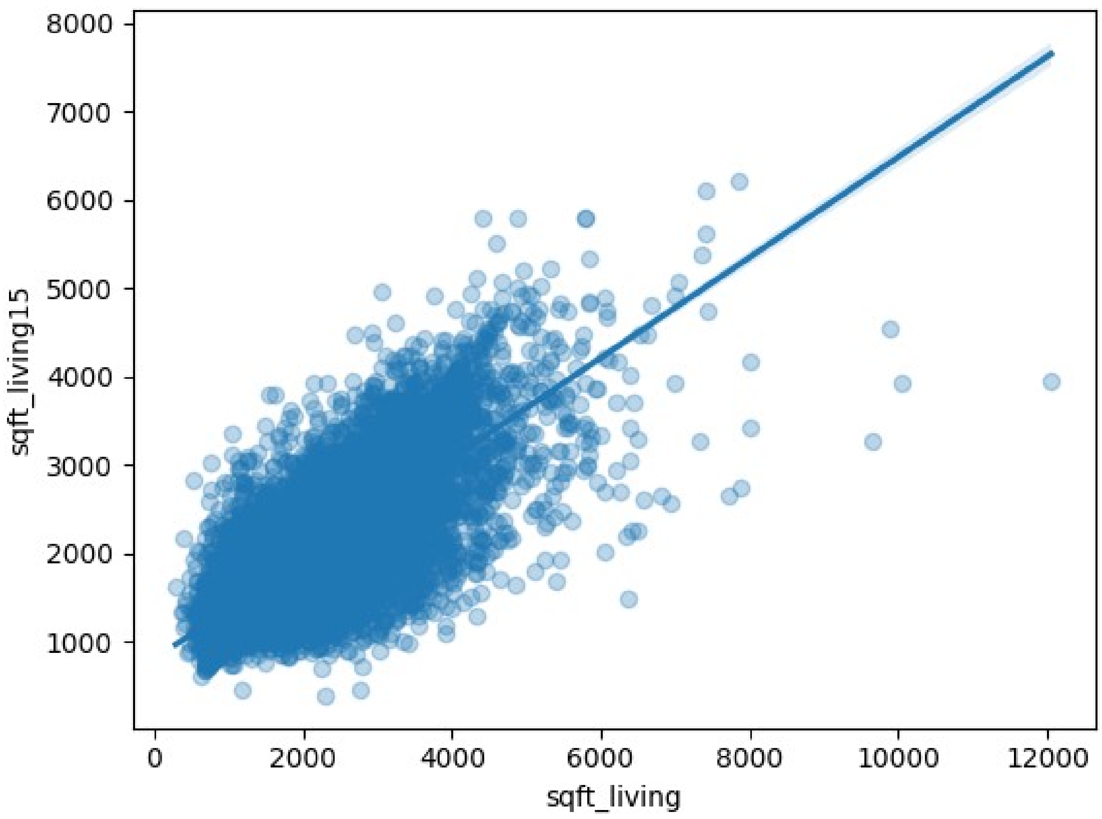


The price values were converted to log to reduce the range of prices that can we represented well on the plot. The graph clearly shows how houses in a certain price range tend to be clustered together. Spatial influences such as these are clearly interpretable from the tabular data. Longitude and latitude together play an important role in determining house prices.

```
plt.figure(figsize=(12,6))

sc = plt.scatter(
    "sqft_living",
    "sqft_living15",
    c=np.log1p(property_data["price"]),
    data=property_data,
    cmap="viridis",
    s=12,
    alpha=0.6
)

plt.colorbar(sc, label="log(Price)")
plt.title("House Size vs Neighborhood Size\n(Color = log(Price))")
plt.xlabel("Individual House Size (sqft_living)")
plt.ylabel("Neighborhood Avg Size (sqft_living15)")
plt.show()
sns.regplot(x="sqft_living",y='sqft_living15', data=property_data,
scatter_kws={"alpha":0.3} )
```



```
<Axes: xlabel='sqft_living', ylabel='sqft_living15'>
```

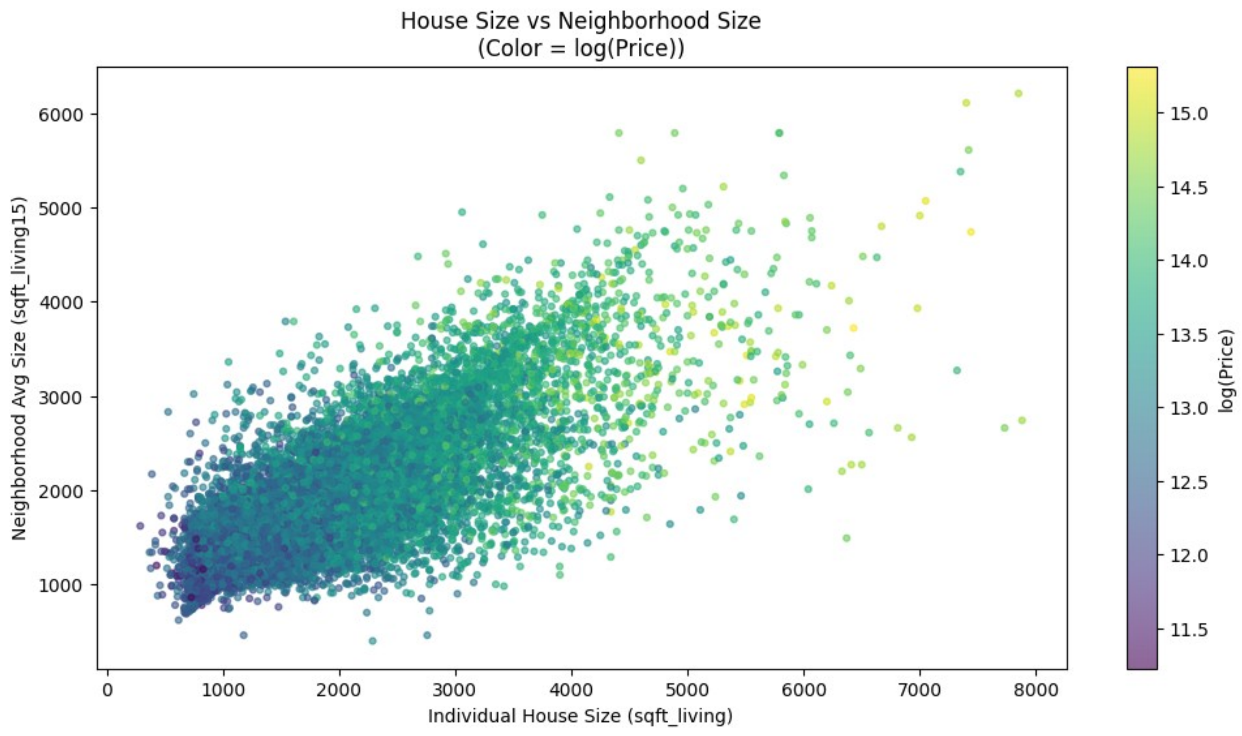


There is a clear trend between the size of houses in an area and the size of the house itself and how they affect the price.Prices increase most strongly in the upper-right region, where both the property and its surrounding neighborhood consist of larger homes. Also, individual houses with a comparatively larger area than their neighbours are all highly priced(yellow in shade).

```
outliers1 = property_data[property_data["sqft_living"] > 8000]

print("Number of outliers:", len(outliers1))
outliers1[["id", "price", "sqft_living", "sqft_living15"]]

no_size_outlier = property_data[property_data["sqft_living"] <=
8000].copy()
```

```
Number of outliers: 6
```

```
plt.figure(figsize=(12,6))

sc = plt.scatter(
    "sqft_living",
    "sqft_living15",
    c=np.log1p(no_size_outlier["price"]),
    data=no_size_outlier,
    cmap="viridis",
    s=12,
    alpha=0.6
)

plt.colorbar(sc, label="log(Price)")
plt.title("House Size vs Neighborhood Size\n(Color = log(Price))")
plt.xlabel("Individual House Size (sqft_living)")
plt.ylabel("Neighborhood Avg Size (sqft_living15)")
plt.show()
```



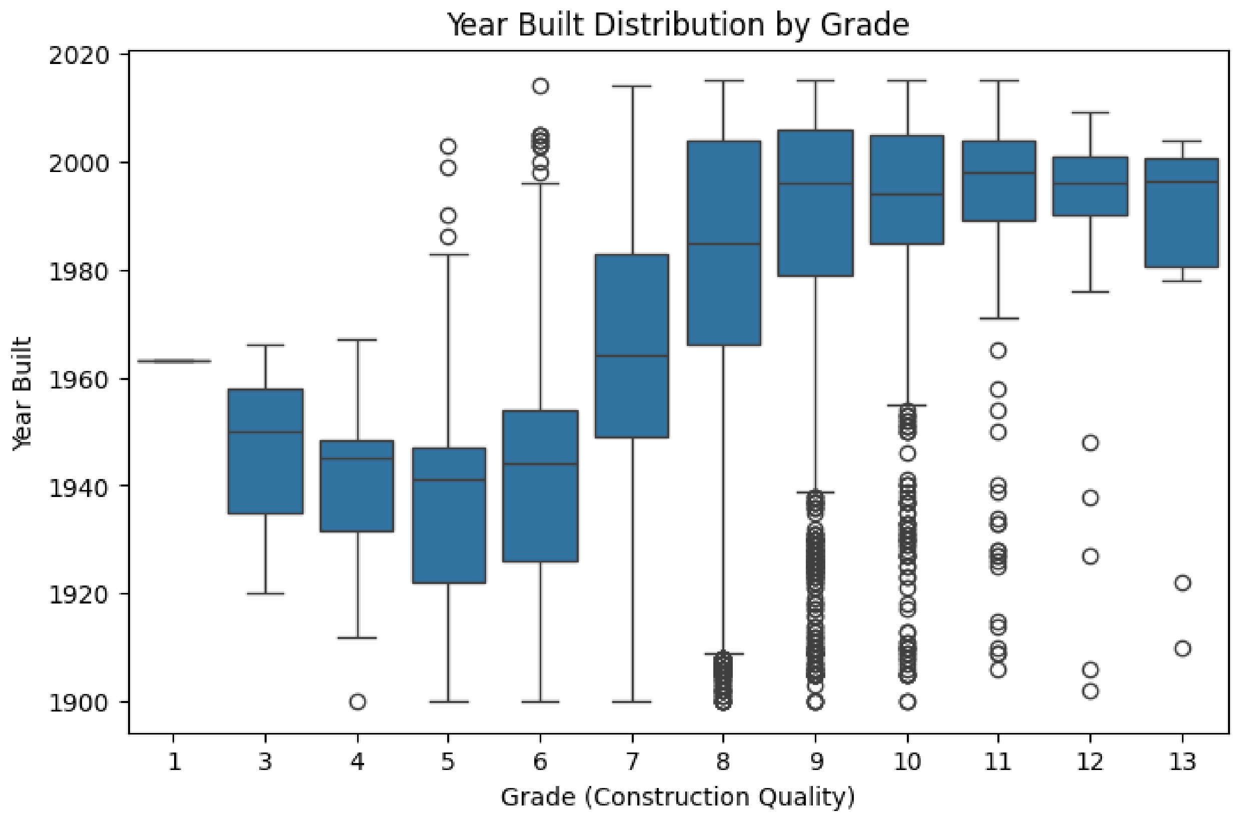
A small number of inputs showed extreme prices compared to the rest of the dataset and hindered woth visual analysis of the data. These are very likely rare, luxury houses which do no represent the geenral trend of proeprty valuation. Also, it can be observed that for a house of a fixed size, the prices are likely to be higher if the house is located in a neighbourhood of comparatively larger houses. Thus 'sqft_living15' is a contributing factor is determining house prices.

```
print(property_data.corr()['grade'].sort_values(ascending=False))

grade            1.000000
sqft_living      0.760925
sqft_above       0.753012
sqft_living15    0.707371
bathrooms        0.664627
price            0.664266
floors           0.457886
yr_built         0.446375
bedrooms         0.351477
view             0.242506
```

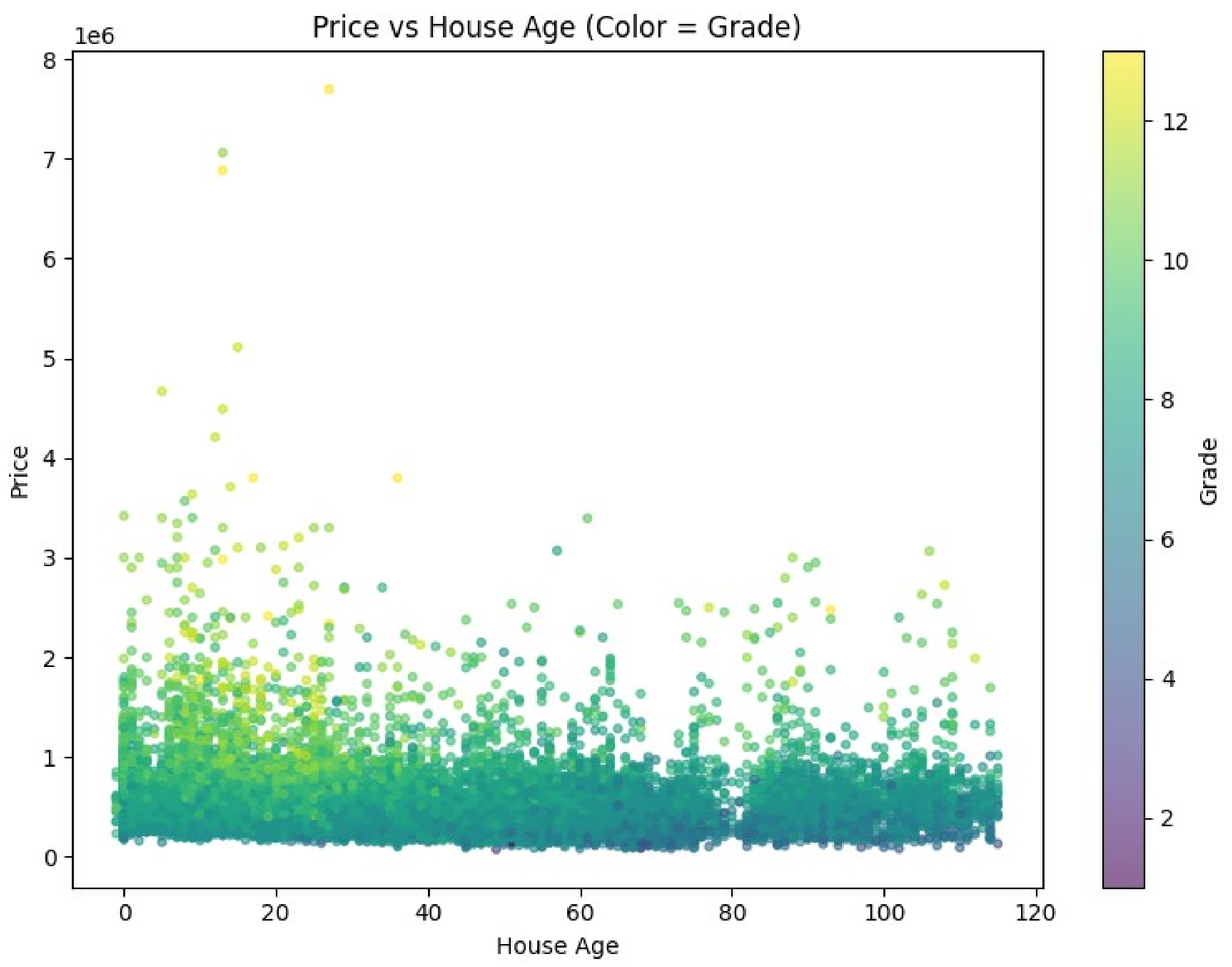
```
long              0.200375
sqft_basement     0.162622
lat               0.117057
sqft_lot          0.113591
sqft_lot15        0.112788
waterfront        0.060923
yr_renovated      0.015810
id                0.010769
date             -0.031094
condition        -0.146829
zipcode          -0.183338
Name: grade, dtype: float64
```

'grade' represents construction quality and architectural design, as defined by the dataset (ranging from poor construction to high-quality custom homes). While grade is strongly correlated with structural attributes such as living area and number of bathrooms, this reflects market tendencies in which higher-quality homes are typically larger and more modern.

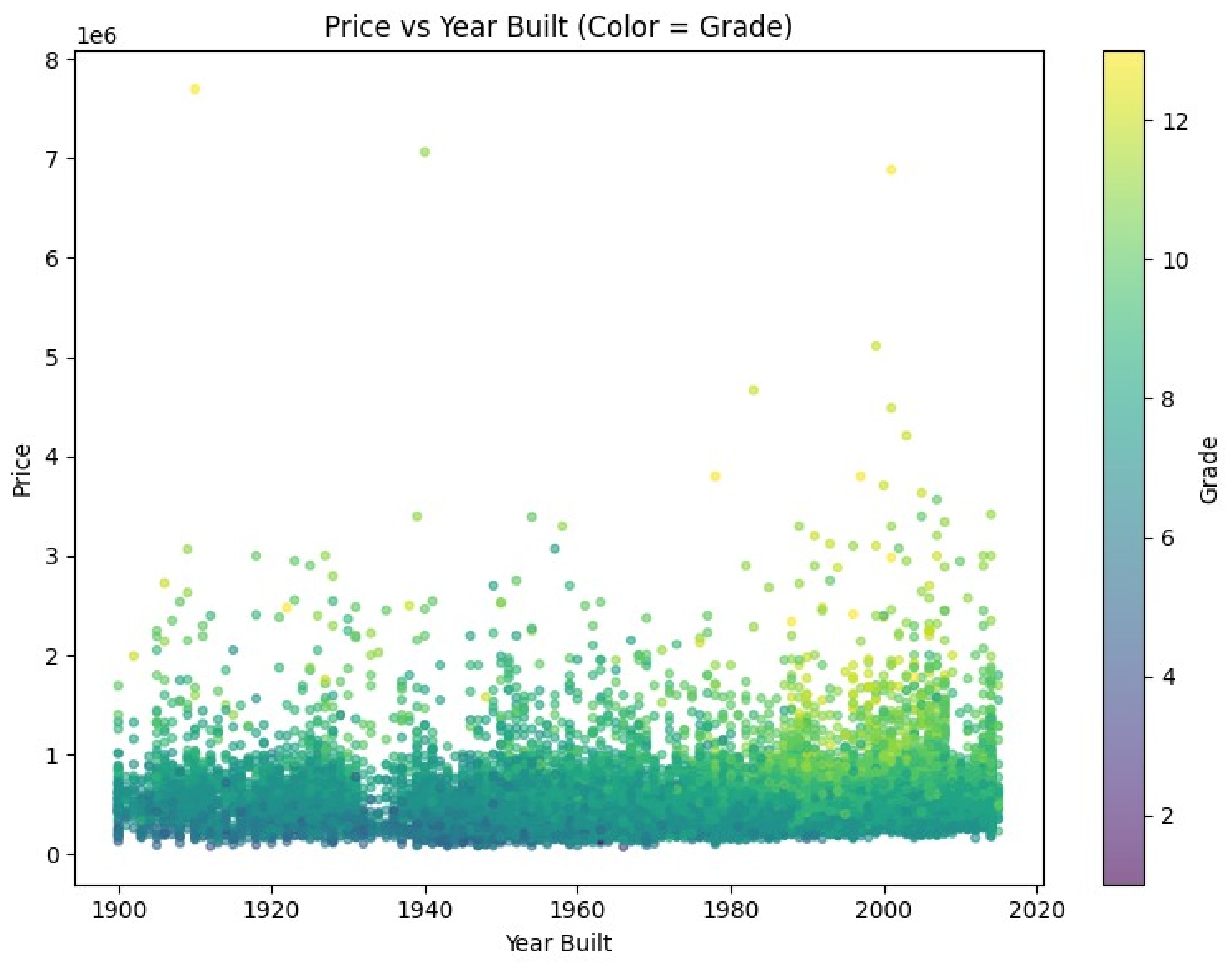
```
plt.figure(figsize=(8,5))
sns.boxplot(x="grade", y="yr_built", data=property_data)
plt.title("Year Built Distribution by Grade")
plt.xlabel("Grade (Construction Quality)")
plt.ylabel("Year Built")
plt.show()
```



```python
# Extract year sold if needed
property_data["yr_sold"] =
pd.to_datetime(property_data["date"]).dt.year

# Effective construction year
property_data["effective_year"] = np.where(
    property_data["yr_renovated"] > 0,
    property_data["yr_renovated"],
    property_data["yr_built"]
)

# House age at time of sale
property_data["house_age"] = property_data["yr_sold"] -
property_data["effective_year"]

import matplotlib.pyplot as plt
import numpy as np

plt.figure(figsize=(8,6))

sc = plt.scatter(
    property_data["house_age"],
    property_data["price"],
    c=property_data["grade"],
    cmap="viridis",
    s=12,
    alpha=0.6
)

plt.colorbar(sc, label="Grade")
plt.title("Price vs House Age (Color = Grade)")
plt.xlabel("House Age")
plt.ylabel("Price")
plt.tight_layout()
plt.show()
```



```
plt.figure(figsize=(8,6))

sc = plt.scatter(
    property_data["yr_built"],
    property_data["price"],
    c=property_data["grade"],
    cmap="viridis",
    s=12,
    alpha=0.6
)

plt.colorbar(sc, label="Grade")
plt.title("Price vs Year Built (Color = Grade)")
plt.xlabel("Year Built")
plt.ylabel("Price")
plt.tight_layout()
plt.show()
```



```
corr_age_grade =
property_data["house_age"].corr(property_data["grade"])
print("Correlation between house_age and grade:", corr_age_grade)

corr_age_grade =
property_data["yr_built"].corr(property_data["grade"])
print("Correlation between yr_built and grade:", corr_age_grade)
```

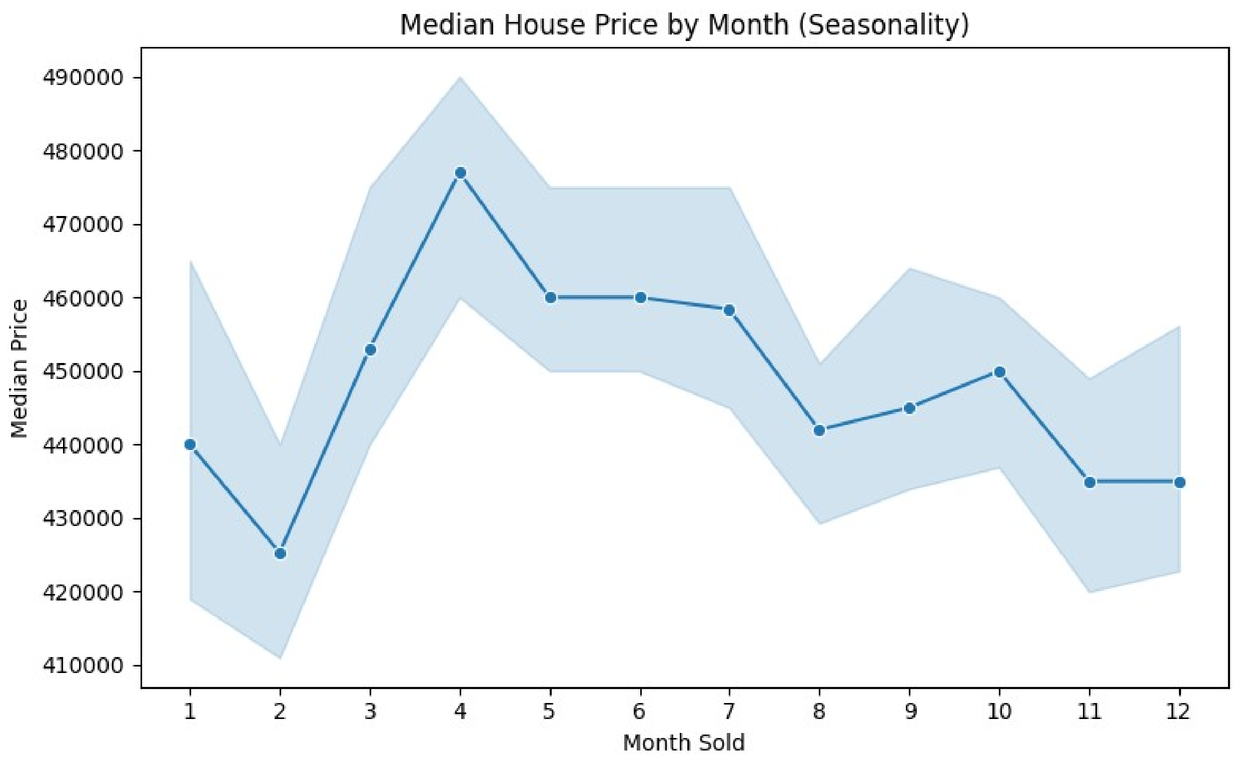
```
Correlation between house_age and grade: -0.4622692707722645
Correlation between yr_built and grade: 0.44637509296731825
```

As can be observed, 'house_age' takes into account the cases when house was renovated as well, giving a more realistic expectation of the 'grade' value of the house. Thus it directly relates to property prices.

```
property_data["date_num"] =
property_data["date"].map(pd.Timestamp.toordinal)

corr_date_price =
property_data["date_num"].corr(property_data["price"])
print("Correlation between sale date and price:", corr_date_price)
```

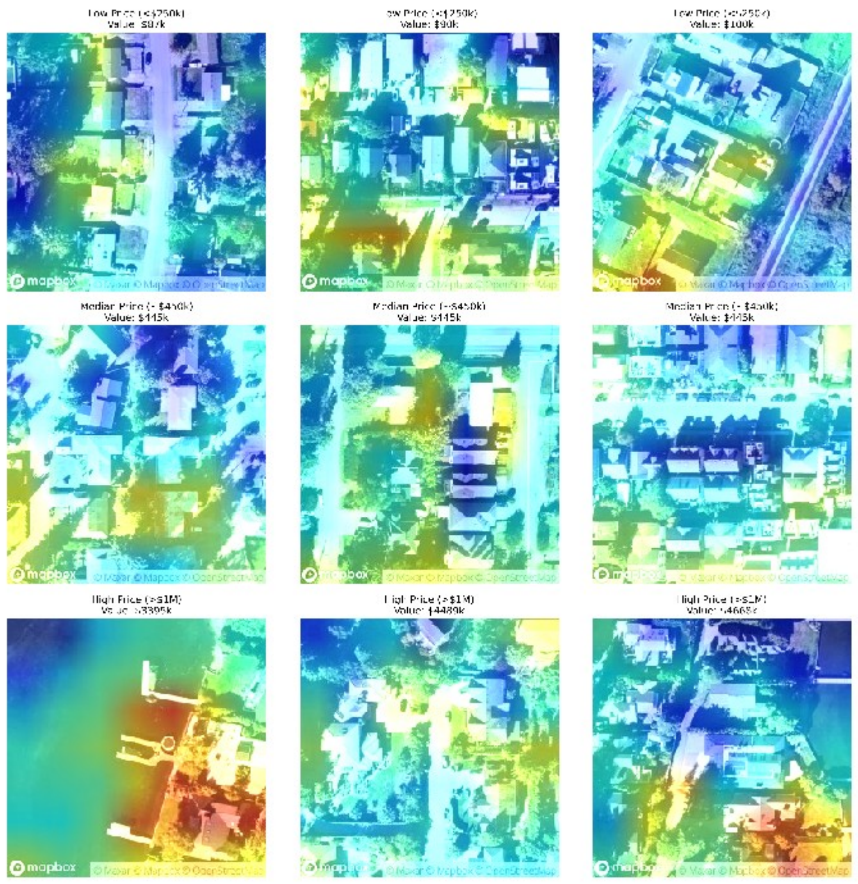
```
Correlation between sale date and price: 0.004309613362903663
```

```
property_data["date"] = pd.to_datetime(property_data["date"])
property_data["month_sold"] = property_data["date"].dt.month
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))

sns.lineplot(
    x="month_sold",
    y="price",
    data=property_data,
    estimator="median",
    marker="o"
)

plt.title("Median House Price by Month (Seasonality)")
plt.xlabel("Month Sold")
plt.ylabel("Median Price")
plt.xticks(range(1,13))
plt.tight_layout()
plt.show()
```



The plot shows median prices of houses on a monthly basis. While slight fluctuations are observed, the overall variation remains limited relative to core structural and neighborhood features. These effects are due to changing market trends, and are not indicative of the intrinsic properties of a house.

```python
from PIL import Image

img = Image.open("/kaggle/input/gradcam-image/gradcam_analysis.png")
plt.figure(figsize=(6, 6), dpi=150)
plt.imshow(img, interpolation="nearest")
plt.axis("off")
plt.show()
```



The first row shows images at a low price range(<100k), second row at prices close to the median(~450k), and third row with houses on the higher end(>1M). This heatmap was taken for the cnn trained on residual method. As a result, the results here focus more on what made the prices of the tabular data off, and how the cnn could fix it. It is looking at subtle environment cues, which are tough to interpret as features because of the way model was trained.

Results

Model Performance Comparison

Model	Input Features	R²	RMSE (₹ / $)	Improvement
Tabular Only (XGBoost)	Structured features only	0.910899	109,884	—
Tabular + Satellite Images (Residual CNN)	Tabular + Image-based correction	0.909921	116,123.98	-0.00169 R²

After multiple trials, it was observed that cnn output degraded the performance of the tabular model in most of the cases. So I considered the Baseline Tabular model to make my final predictions.

Previous Approaches

CNN Feature Extraction + PCA

Idea:

We initially used a pretrained CNN (EfficientNet-B0) to extract high-dimensional image embeddings for each property. These embeddings were then:

Reduced using Principal Component Analysis (PCA), and

Concatenated with tabular features before training a regression model.

Outcome:

While this approach provided a compact representation of visual features,

The fused model either showed marginal improvement or degraded performance compared to the tabular baseline.

Reason:

The extracted features were high-level and generic (ImageNet-trained), capturing broad textures and shapes but lacking strong task-specific relevance for property pricing. PCA further compressed this information, often discarding weak but meaningful environmental cues

Direct CNN Price Prediction + Late Fusion

Idea:

We next trained the CNN to predict property prices directly from satellite images. The CNN predictions were then combined with the tabular model using a late-fusion strategy (e.g., weighted averaging).

Outcome:

The CNN alone performed significantly worse than the tabular model.

Fusion either provided minimal gains or introduced noise, sometimes reducing overall R^2.

Reason:

Satellite images alone do not contain sufficient detail about interior attributes, construction quality, or numerical factors such as square footage and grade. As a result, the CNN struggled to learn absolute price levels reliably.

Residual Learning (Final Approach)

Motivation:

Rather than forcing the CNN to learn the entire pricing function, we reframed its task:

Let the tabular model predict price, and let the CNN learn only what the tabular model gets wrong.

Advantages Observed:

The CNN focuses on hard examples and edge cases instead of relearning obvious patterns.

Visual information is used only where it adds value, improving stability and generalization.

Grad-CAM analysis revealed that the CNN attends to broad environmental context rather than isolated objects, aligning with the hypothesis that imagery provides neighborhood-level cues.

This residual framework consistently outperformed earlier fusion methods to some extent.