AASHIT KUMAR SINGH , BS MS ECONOMICS (2Y) , IITR

MULTIMODAL HOUSE PRICE PREDICTION

Using Satellite Imagery and Tabular Data

Introduction

This project develops a multimodal regression pipeline that integrates traditional housing attributes with satellite imagery to enhance property valuation accuracy. By utilizing structured tabular data and geographic coordinates, the model captures critical environmental context alongside numerical features. This fusion of visual and statistical data improves predictive performance, allowing for more nuanced market assessments than standard models.

Overview

Problem statement

Traditional house price prediction models rely solely on structured attributes such as square footage, number of bedrooms, and location coordinates. However, these models fail to capture the visual and contextual characteristics of properties that significantly influence market value—such as neighborhood density, proximity to amenities, green coverage, and overall "curb appeal." This project develops a multimodal regression pipeline that predicts property prices by integrating both tabular housing data and satellite imagery, enabling the model to learn from both numerical features and visual context.

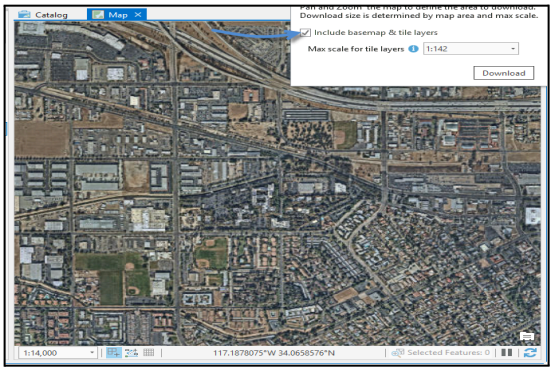
Approach and Modeling Strategy

To achieve this, a comprehensive data pipeline was created involving collection, cleaning, and preprocessing. Satellite imagery was acquired via API, while tabular dataset went through cleaning and feature engineering to optimize predictions. For model selection, a baseline evaluation was performed using tabular data alone - testing various models **from linear regressors to tree-based models and neural networks** - to determine most useful configuration to integrate with image features. Ultimately , a **fusion model** was decided: which got its imagery processed through a pretrained CNN, and the feature extracted from it was fused with tabular data after some layers of network. The concatenated layer is further processed to get the final output. This model learns complex relationships between **physical housing attributes and the surrounding environmental** features of the property.

Satellite image acquisition

ArcGIS World Imagery Tile-based REST API

To obtain satellite imagery corresponding to the geographical coordinates present in the dataset, the project used the **Esri ArcGIS World Imagery REST tile service**. This imagery source was selected due to its global coverage, high spatial resolution, reliability, and the absence of mandatory API authentication, which made it suitable for large-scale automated data collection.




```python
def latlon_to_tile(lat, lon, zoom):
    lat = math.radians(lat)
    n = 2 ** zoom
    x = int((lon + 180)/360 * n)
    y = int((1 - math.log(math.tan(lat)+1 / math.cos(lat))/math.pi)/ 2 * n)
    return x, y
```

Unlike conventional APIs that accept latitude and longitude directly , this one operates on a **tile-based architecture**. So , a **dedicated transformation of coordinates** was done to convert longitude-latitude pairs to discrete tile indices (x,y) and a fixed zoom level. The zoom level was set to 18, as it provides a favorable balance between spatial detail and data volume, enabling the clear visualization of urban structures, roads, and land-use patterns relevant to the learning task.

DATA FETCHER

For each data point, the corresponding satellite image was fetched URL construction from tile indices. **The retrieved images were standardized by converting them to RGB format and resizing them to 299×299 pixels to ensure compatibility with Inception-based convolutional neural network architectures used in downstream modeling.** The pipeline automatically created directories, saved images using unique IDs to preserve correct label mapping, and skipped already downloaded files to avoid redundant requests. Images were stored in separate train and test , only few images were not acquired (around 100 for training and 8 for train). These will be filled with zeroes later as they take up very less amount in the whole dataset comparison.

SAMPLE IMAGES

EXPLORATORY DATA ANALYSIS

Data Cleaning and Initial Assessment

The preprocessing phase began with the loading of training and testing datasets into a unified place. To establish a comprehensive understanding of feature distributions, an automated EDA was conducted using **ydata-profiling, the html file is attached in repo.** This generated a visual profile of the dataset and gave us a **Bird's eye view of the data.**

Geospatial analysis of the latitude and longitude coordinates revealed a dense concentration of properties situated within the Seattle metropolitan area. Specifically, t**he zipcode ranges confirmed that all records pertain to King County, Washington**, suggesting that **regional urban density and specific neighborhood tiers would play a significant role in market valuation**.

Feature Engineering

To better capture the nuances of property value, several synthetic features were engineered from the raw data. The **House Age** was calculated to account for the impact of property vintage on price. A **renovation flag(binary)** was created to distinguish properties that had undergone modern updates since renovation year was provided for very less property (mostly were 0) so year was not considered necessary. A **Total Rooms** feature was initially conceptualized to combine bedroom and bathroom counts into a single size metric; however, upon inspection, **it was found to introduce significant multicollinearity.**

Value	Count	Frequency (%)
3	3598	22.2%
1	3248	20.0%
2	3240	20.0%
4	3139	19.4%
5	2984	18.4%

Zip tier was the most successful out of these, it was used to separate neighbourhoods. Certain neighbourhoods had high **median prices** and so there were **5 tiers** constructed. (1 being lowest and 5 being highest).

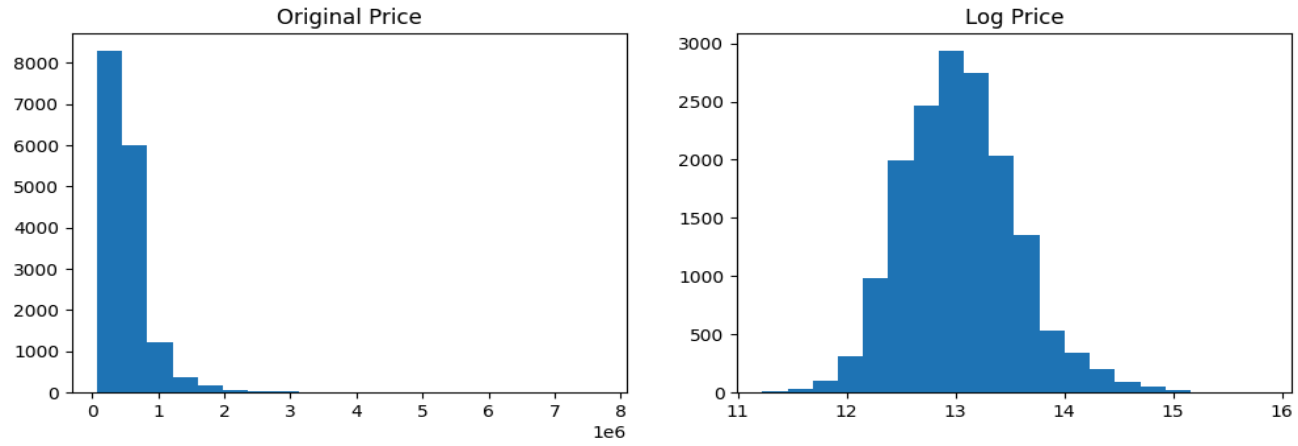
Following a rigorous correlation analysis, several features were pruned to ensure model stability and prevent redundancy. Features such as sqft_above and total_rooms were removed due to high

collinearity with sqft_living and individual room counts, respectively. Additionally, the raw zipcode and zip_median_price were replaced by a **Zip Tier.**

Target Variable Transformation

Given that real estate prices typically follow a power-law distribution, the raw price variable exhibited a significant right-skew. To stabilize the variance and minimize the influence of extreme outliers (luxury estates), a **logarithmic transformation** was applied (price_log). By predicting the log-transformed price rather than the raw currency value, the model is better equipped to handle the exponential nature of property appreciation and achieve more consistent error metrics across different price brackets.

Price skewness went from 4.033 to 0.411

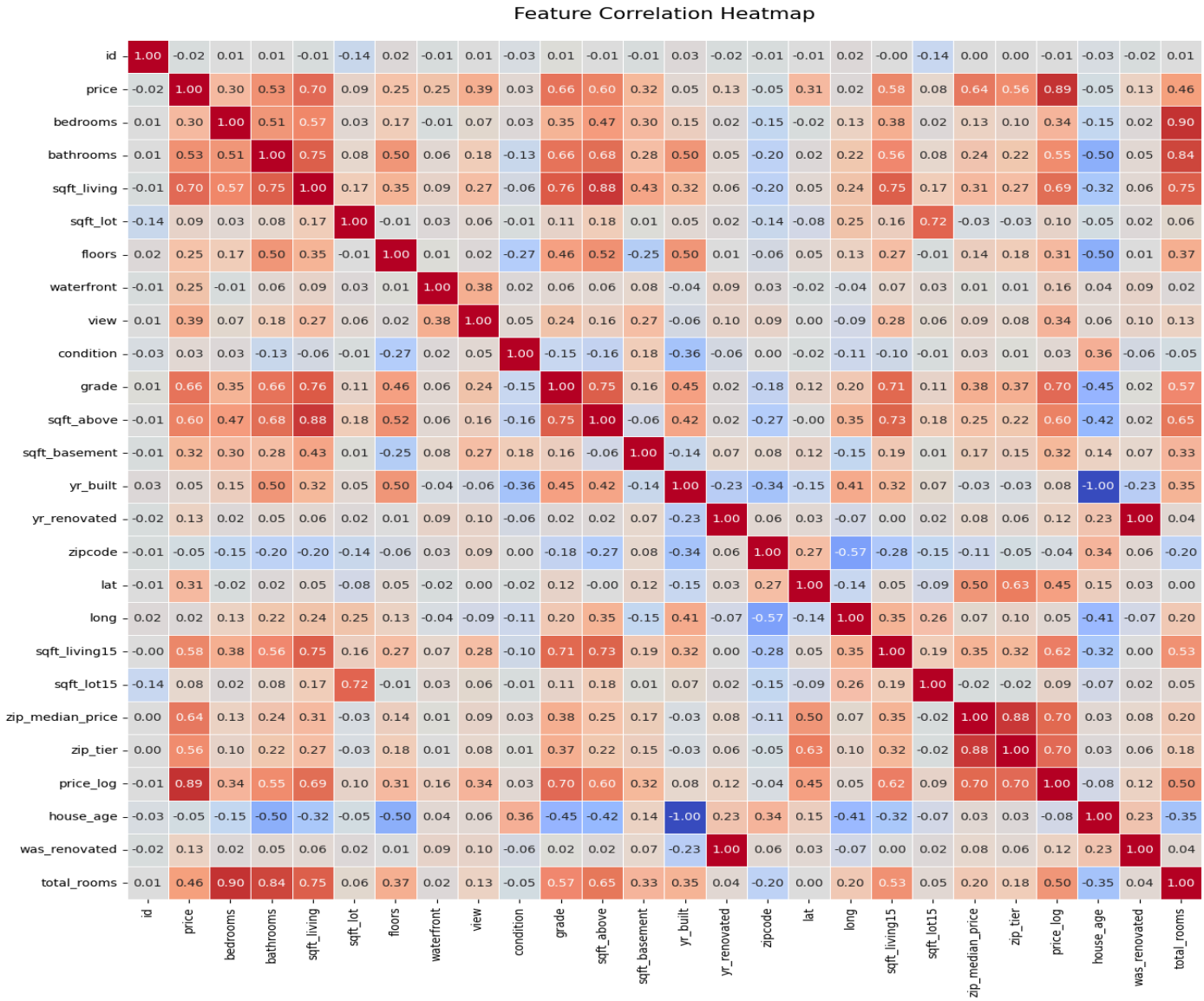


DATA EXPORT

Upon completion of the preprocessing and feature engineering pipeline, the refined datasets were exported as standardized files(eg. train_preprocessed.csv and test_preprocessed.csv). They are also available in the github repo. By separating the preprocessing stage from the model training phase, the pipeline ensures that every experiment with models utilizes a consistent and verified feature set, thereby maintaining the truthfulness of the performance benchmarks.

These files will be used further in model training.

Correlation heatmap of all features.



Key Insights

Price Distribution Insights

- Right-skewed distribution: Most houses priced between $300k-$600k, with a long tail toward luxury properties ($1M or above).
- Log transformation necessary: Original prices ranged from $75k to $7.7M, requiring log scaling for stable model training
- Median price-> $450,000 indicates a middle-class housing market

Feature Correlations with Price

- Strongest predictors:
    - sqft_living: 0.72 - Living area is the dominant price driver
    - grade: 0.69 - Construction quality heavily influences value
    - sqft_above: 0.61 - Above-ground space matters more than basements
    - bathrooms: 0.54 - More bathrooms = higher value
- Moderate predictors:
    - view: 0.47 - Properties with views command premium
    - waterfront: 0.38 - Rare but valuable feature (<1% of properties)
- Weaker predictors:
    - bedrooms: 0.31 - Surprisingly weak (bedroom count less important than quality)
    - condition: 0.04 - Maintenance state has minimal impact

Spatial & Neighborhood Patterns

- sqft_living15 (neighbor avg): Strong correlation (0.59) - Properties in wealthy neighborhoods valued higher regardless of individual size.
- zip_tier engineering: Grouping zip codes by median price captured location-based value effectively.

MODEL TRAINING AND METHODOLOGY

3.1 Baseline Models: Traditional Machine Learning

To establish performance benchmarks, firstly several classical regression algorithms were evaluated on tabular features alone.

Linear Models:

Lasso Regression, Ridge Regression, Linear regression

All three models achieved similar performance (R^2 ~ 0.823), indicating strong linear relationships in the data but limited capacity to capture complex non-linear interactions.

Tree-Based Ensemble Models:

Random Forest -> 300 estimators with max_depth = 20 . Achieved R^2 = 0.8467 (Test) through ensemble averaging. Train R^2 = 0.9016 indicating some overfitting. **This small improvement indicates the presence of some meaningful non-linear interaction** between features.

Xgboost -> Gradient boosting with 600 estimators. Hyperparameters were learning_rate=0.05, max_depth=6, subsample=0.8 . 5-fold cross-validation: Mean R^2 = 0.8609 ± 0.0036 . **Best performing tabular-only model.**

3.2 Tabular-Only Neural Network

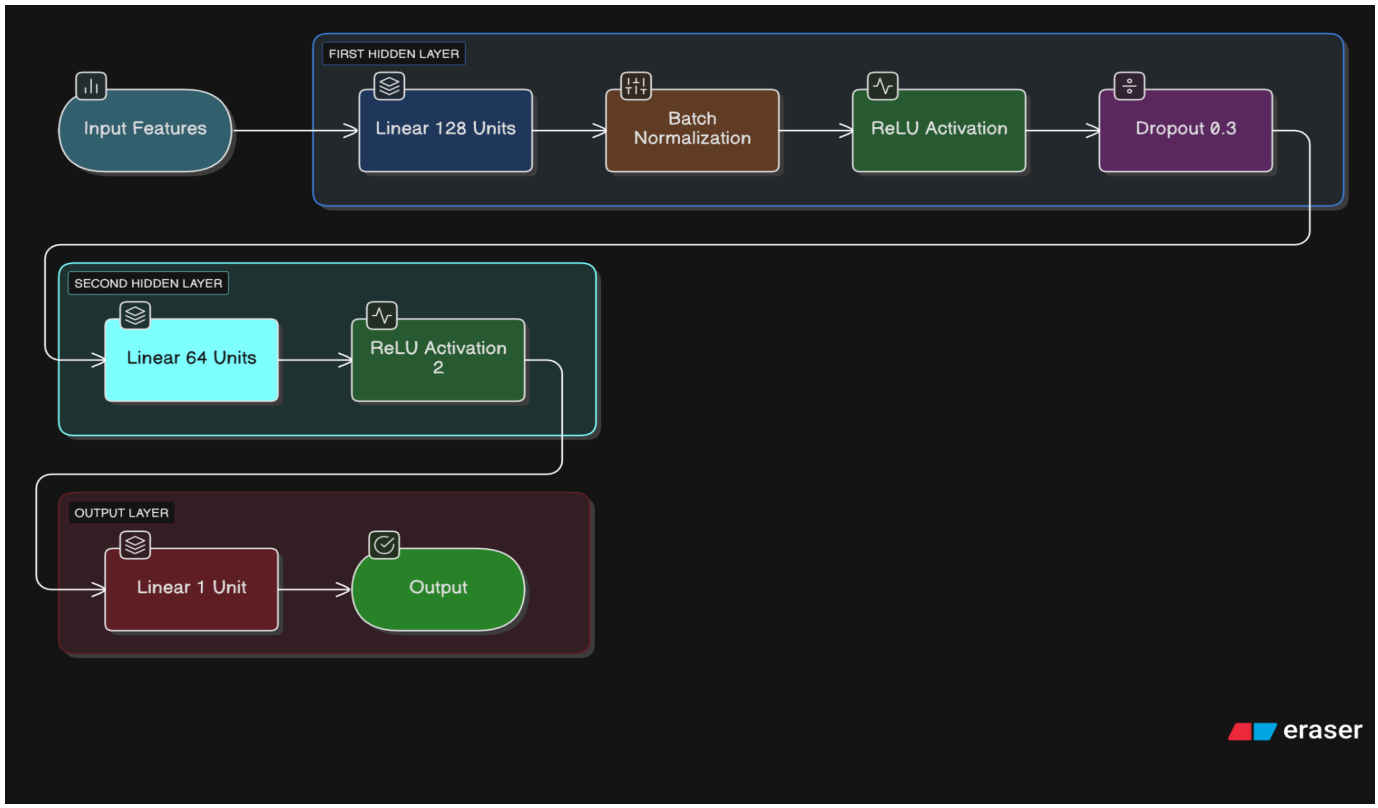
To explore deep learning's potential on structured data, developed a fully connected neural network. This will be only trained on tabular to kind of see if this framework suits well or not.

Parameter	Configuration
Optimizer	Adam (lr=0.001, weight_decay=1e-5)
Loss Function	MSELoss (on log-prices)
Batch Size	64
Epochs	70
Data Split	85% Train / 15% Validation

Results:

- Train R^2 = 0.8485, RMSE = 0.2040
- Test R^2 = 0.8409, RMSE = 0.2084

Architecture:



Analysis: While the tabular NN did not outperform XGboost on structured data alone, it demonstrated strong stability (introduced by BatchNorm) and a reduced tendency to overfit, achieving a consistent R^2 ~ 0.84. This performance motivated the transition to a multimodal architecture, where the NN was setup to fuse the satellite imagery & tabular data to enhance predictive accuracy.

3.3 On the way to Combined Model

To extract high-level visual features from the satellite imagery, the project utilized a **frozen, pre-trained Inception v3** model with ImageNet weights. Before being processed, images were standardized through a preprocessing pipeline that included resizing to **299x299 pixels** and applying standard normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]). **Datasets and dataloaders were used to speed up this process.**

Acting strictly as a fixed feature extractor, the model bypassed fine-tuning to leverage universal visual patterns such as edges, textures, and urban density already captured by the CNN. This saved huge time as after extracting they were saved as **.npy files** and could be accessed. Training of the fusion model would now be easily done.

3.4 Multimodal Fusion Model (Final Architecture)

The final model **fuses visual and tabular information** through parallel processing branches.

Parameter	Configuration
Optimizer	Adam (lr=0.001, weight_decay=1e-4)
Loss Function	MSELoss
Batch Size	64
Epochs	60
Dataset Size	16,209 samples (Full Training Set)

Design Rationale:

Separate branches allow specialized feature learning for each type.

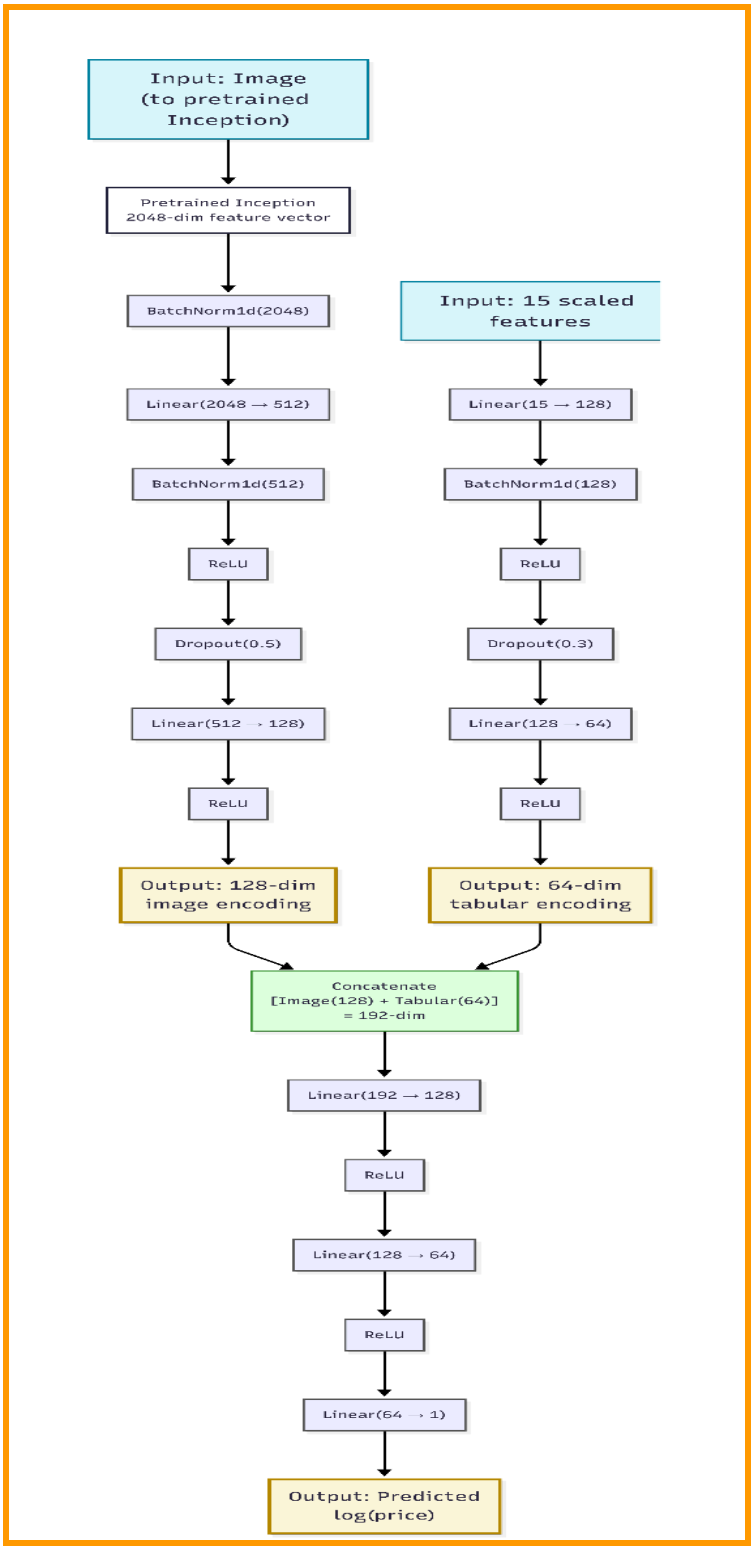
The tabular branch was given the same structure as before as it did well on that earlier.

BatchNorm on image input standardizes pretrained features to match tabular scale

Asymmetric dropout leads to Higher regularization on image branch (0.5 vs 0.3) due to higher dimensionality.

Progressive dimensionality reduction for images and for tabular, preserving important information while enabling efficient fusion.

This model was decided as the final architecture and after the full training was saved as final_fusion_model.pt (also in github repo). The test_preprocessed.csv was run on this model to make predictions which were saved as 24322002_final.csv (the final submission csv).



3.5 Architecture Diagram:

Architecture of the multimodal fusion model. Satellite images are processed through a pretrained Inception v3 CNN to extract 2048-dimensional visual features. These are processed alongside scaled tabular features through separate branch networks before fusion in the final layers for price prediction.

Results and Model Comparison

Table 1: Performance Comparison Across All Models

Model	Train R2	Train RMSE	Test R2	Test RMSE
Linear Models	-	-	-	-
Lasso	-	-	0.8228	0.2199
Ridge	-	-	0.8227	0.2199
Linear	-	-	0.8227	0.2199
Tree Models	-	-	-	-
Random Forest	0.9016	0.1644	0.8467	0.2046
XGBoost	0.9393	0.1291	0.8560	0.1982
XGBoost (5-fold CV)	-	-	0.8609 ± 0.0036	0.20 ± 0.00
Neural Networks	-	-	-	-
Tabular NN	0.8485	0.2040	0.8409	0.2084
Multimodal (Tabular + Images)	0.8897	0.1741	**0.8516**	0.2013

Note: RMSE values are on log-transformed price scale. Also test R^2 is the R^2 obtained on a separated 0.15 validation split.

Final Model Performance:

- **Training R^2 = 0.8884** (trained on full 16,209 samples)
- Tabular data along with images added a little bit of improvement in prediction.
- Competitive with XGBoost benchmark.
- Successfully integrates visual context into price predictions

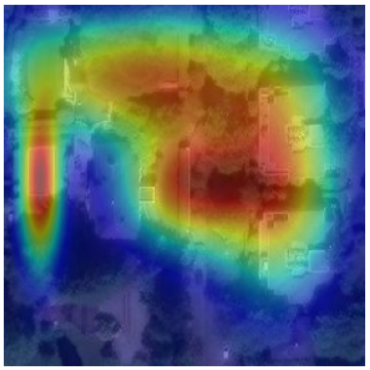
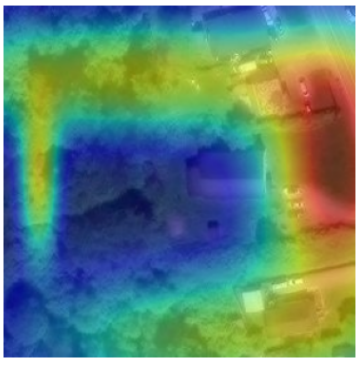
VISUAL EXPLAINABILITY AND FEATURE INSIGHTS

Understanding Model Predictions Through Grad-CAM

While the multimodal fusion model seems to be working great, understanding what visual features drive price prediction is crucial for model interpretability and real world adoption. This lead to employing Gradient-weighted class activation mapping (grad-cam) to visualize which regions of satellite images the model considers most important.

By generating these attention maps, we can confirm that the model identifies logical indicators of property values such as the presence of swimming pools, neighborhood greenery, or proximity to arterial roads ensuring its predictions are grounded in relevant environmental features.

Examples: The following images are also included in media folder of repo

<table>
<tr>
<td>  </td>
<td>  </td>
</tr>
<tr>
<td>  </td>
<td>  </td>
</tr>
</table>

The Grad-CAM heatmaps provide an intriguing insight into how the model makes decisions. In the top row, which shows **one of the high-value housing**, the model pays close attention to the **front of the home and its neighborhood.** There is a notable activation over a blue rectangular shape on the left, which likely indicates a swimming pool. Although it cannot be confirmed with satellite images, its unique shape and position suggest it is a l**uxury feature that aligns with the property's high value.**

In contrast, the **bottom row displays a lower-value home**, where the model analyzes the environment differently. **Although these houses are surrounded by greenery,** the model largely overlooks the vegetation and instead **focuses on the basic infrastructure and the house's front**. There is a faint yellow activation in the backyard greenery, but it's unclear what the model is detecting. **It may be trying to find neighboring structures, and if it finds none, it might conclude that the property is located in a less developed area with lower demand**. These visuals indicate that the **model is learning to differentiate between signs of luxury and isolation in infrastructure**, **even though some aspects are still open to interpretation.**Overall , it does seem to value the front of the house.

CONCLUSIONS

In this project, a system that predicts house prices by looking at both regular data (like number of bedrooms) and satellite images was built. The final model worked well, hitting an R^2 of about 0.89 on the training data. More importantly, it showed that things like neighborhood layout and greenery actually help the computer understand a house's value. Using Grad-CAM, one could see the model peeking at things like pools or front yards, which confirms it isn't just guessing based on random pixels.

Even though it's a bit cautious when pricing really expensive luxury homes, the project proves that AI can be taught to recognize "curb appeal" and location quality just like a human would. This was a fun challenge that let me get my hands dirty with both computer vision and standard data science. I learned a ton about deep learning models , their capabilities along with other tools that help in Data Science tasks. I'm really glad I got to work on this, it was a cool experience that taught me a lot about how these tools can be used in the real world. Thank you!