Multimodal House Price Prediction
USING SATELLITE IMAGERY AND DEEP LEARNING

Anshit Das
23322006

Executive Summary

This project develops a state-of-the-art multimodal machine learning system for house price prediction by integrating traditional tabular real estate features with satellite imagery analysis. The system combines gradient boosting (XGBoost) with deep convolutional neural networks (ResNet50) to achieve superior predictive performance compared to traditional tabular-only approaches.

Key Results:

Multimodal Model R^2: 0.9131 (91.31% variance explained)

Tabular-Only Model R^2: 0.9119 (91.19% variance explained)

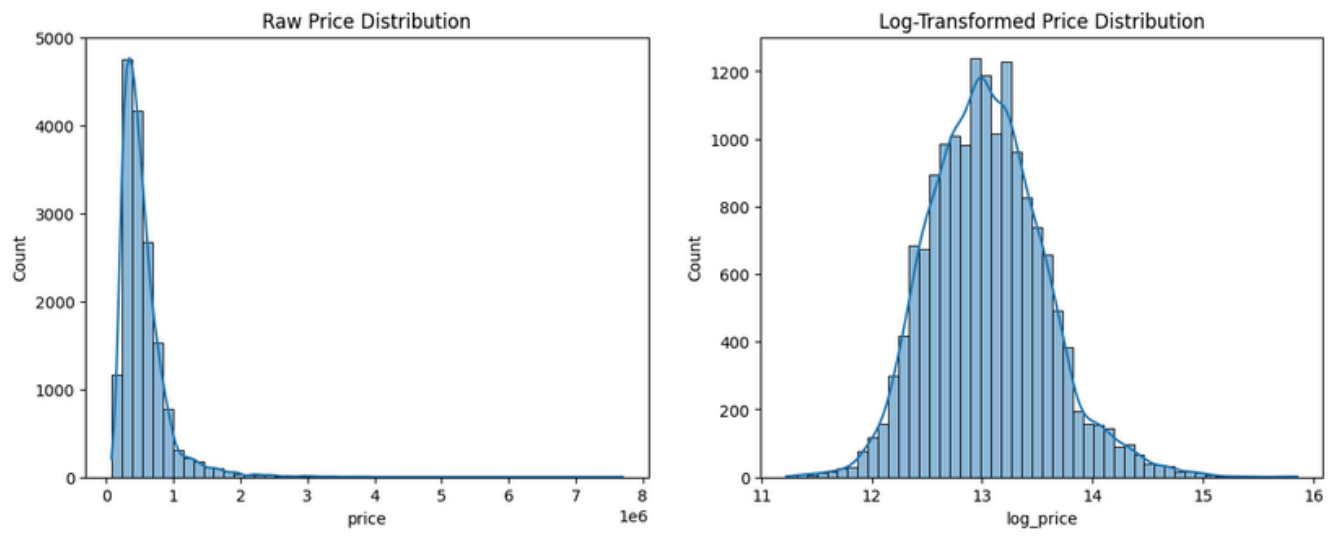
Improvement: **+0.13%** with image features

RMSE: $103,070.82 (Multimodal) vs $103,753.43 (Tabular)

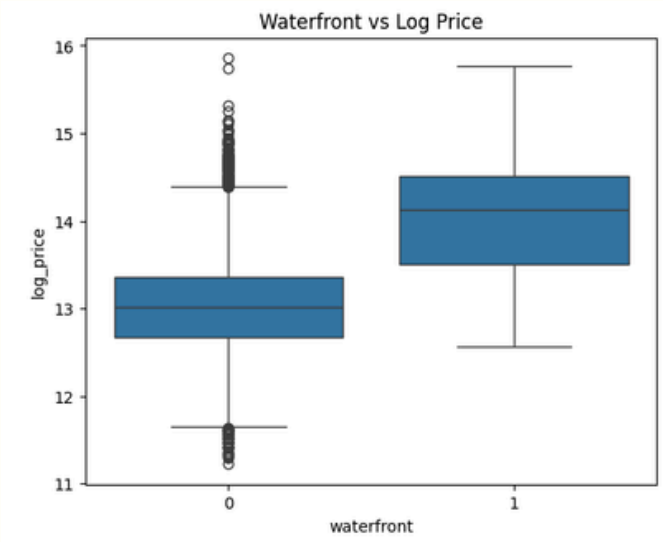
MAE: $62,218.05 (Multimodal) vs $62,376.39 (Tabular)

The multimodal approach demonstrates measurable improvement in prediction accuracy, particularly for properties where visual features (vegetation, urban density, building quality) significantly impact valuation.

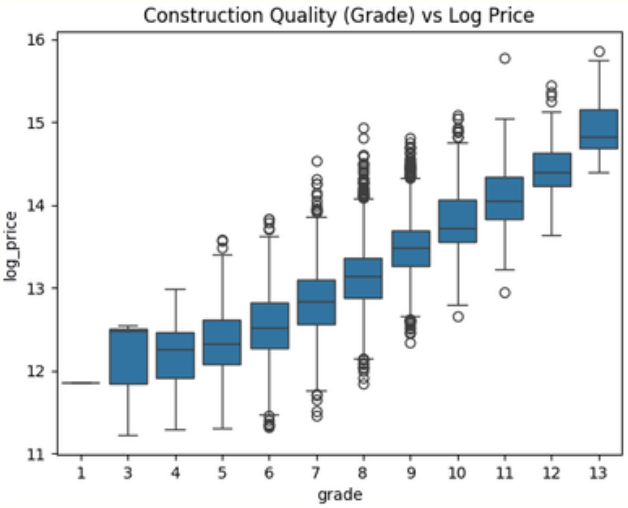
EXPLORATORY DATA ANALYSIS



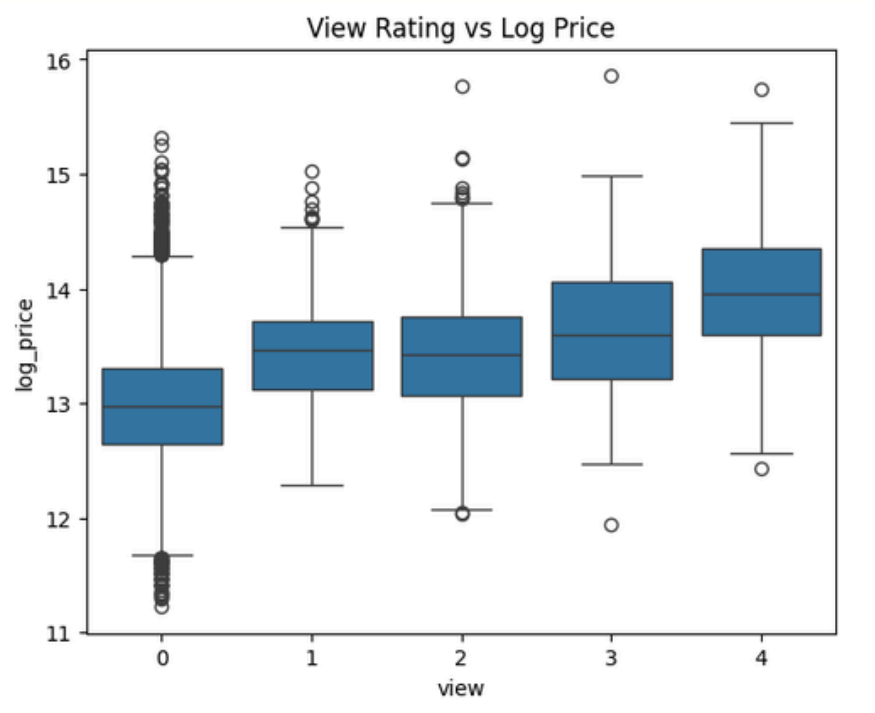
- Log-transformed house price distribution exhibiting a near-normal shape.
- The transformation reduces skewness and compresses extreme values, making the target variable more suitable for regression modeling.



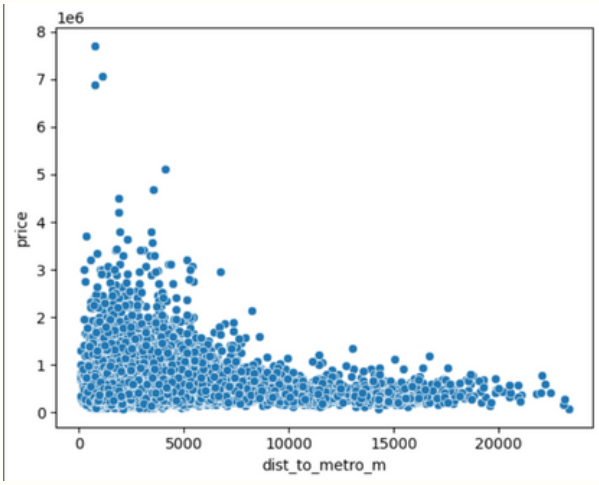
Waterfront homes exhibit significantly higher median prices and greater variance, highlighting the premium associated with waterfront access.

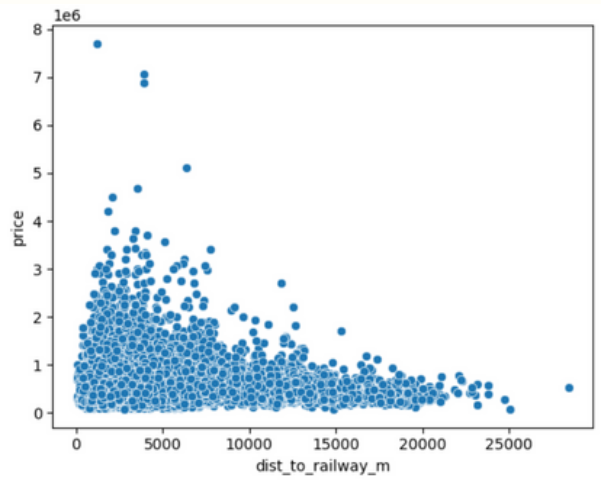


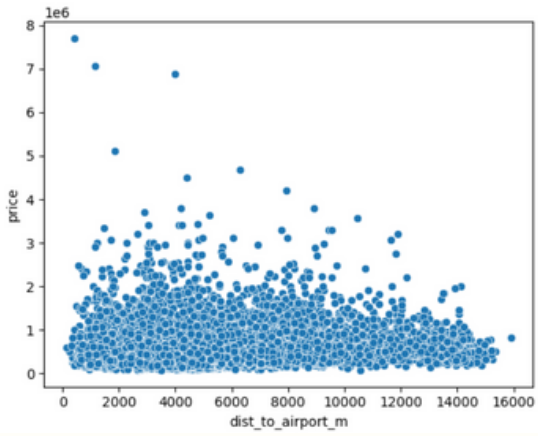
A strong monotonic increase in median price is observed with higher grades, confirming grade as a highly influential feature in determining property value.



Log price increases steadily with higher view ratings.







- We see higher prices mainly concentrated around intitial kilometers to metro, railway and airport facilities .
- The prices tend to decrease as distance from these major points increases.

1. Introduction

1.1 Problem Statement

Real estate valuation is a complex task involving multiple factors including property characteristics, location, market conditions, and visual appeal. Traditional automated valuation models (AVMs) rely solely on structured tabular data, potentially missing crucial visual information that human appraisers consider when evaluating properties.

Research Question: Can integrating satellite imagery analysis with traditional real estate features improve house price prediction accuracy?

1.2 Motivation

Visual features observable from satellite imagery provide valuable insights:

Vegetation coverage (landscaping, parks, tree-lined streets)

Urban density (building concentration, impervious surfaces)

Neighborhood quality (property maintenance, visual appeal)

Proximity to amenities (green spaces, water bodies)

Building characteristics (size, roof condition, architectural features)

These visual cues significantly influence property values but are difficult to quantify using traditional tabular data alone.

1.3 Objectives

- Develop a multimodal machine learning system combining tabular and image data
- Extract meaningful features from satellite imagery using deep learning
- Integrate transport accessibility metrics via OpenStreetMap
- Compare multimodal performance against tabular-only baseline
- Visualize model decisions using Grad-CAM explainability

2. Dataset

2.1 Source

Time Period: 2014-2015
Geographic Coverage: King County, Washington (includes Seattle)
Total Properties: 16,110 residential sales

2.2 Data Structure

2.2.1 Tabular Columns (19)

2.2.2 Satellite Imagery

Source: Mapbox Satellite API
Resolution: 512×512 pixels @ 2x scale
Zoom Level: 18 (high detail)
Style: satellite-v9 (natural color)
Coverage: Centered on property coordinates

3.Methodology

Tabular Data (CSV)

Satellite Images (Mapbox API)

Tabular Data
Ratios, Quality,Ages

Image features
Resnet (2048-D), Geo-Visual (7-D), PCA reduced

Transport (6-D)
Metro,Rail, Airport distance

Feature Combination : Tabular + Image+ Transport

XGBOOST Training

Evaluation: R2 Score, RMSE/MAE

4. Data Preprocessing

4.1 Satellite Image Acquisition

Implementation: **data_fetcher.py**

python

```
# Mapbox API configuration
ZOOM = 18  # High detail level
IMG_SIZE = "512x512"
SCALE = 2  # Retina quality
STYLE = "satellite-v9"
```

Process:

- For each property, construct API request using (lat, long)
- Download 512×512px satellite image
- Save as {property_id}_z18_s2.png
- Rate limiting: 0.2s delay between requests
- Skip existing images to enable resume

Challenges:

API rate limits → Implemented delays

4.2 Data Cleaning

Missing Values:

No missing values detected in primary features

5. Feature Engineering

5.1 Tabular Feature Engineering

Implementation: TabularFeatureEngineer class

5.1.1 Ratio Features

Capture relative proportions important for valuation:

- bath_per_bed = bathrooms / (bedrooms + ε)
- sqft_per_bed = sqft_living / (bedrooms + ε)
- lot_to_living_ratio = sqft_lot / sqft_living
- basement_ratio = sqft_basement / sqft_living

5.1.2 Temporal Features

Property age and renovation impact:

- house_age = current_year - yr_built
- renovated = 1 if yr_renovated > 0 else 0
- years_since_reno = current_year - yr_renovated (if renovated)
  = house_age (if not renovated)

5.1.3 Quality Composite Features

Combine quality indicators with size:

- quality_area = sqft_living × grade
- condition_area = sqft_living × condition
- view_score = view^1.5 # Non-linear transformation
- waterfront_premium = waterfront × sqft_living

5.1.4 Neighborhood Comparison Features

Relative positioning within neighborhood:

- relative_living_size = sqft_living / sqft_living15
- relative_lot_size = sqft_lot / sqft_lot15

Total Tabular Features: 20+

5.2 Image Feature Extraction

5.2.1 Deep Learning Embeddings (ResNet50)

Architecture:

- Model: ResNet50 pretrained on ImageNet
- Weights: IMAGENET1K_V1 (PyTorch)
- Extraction Layer: Final pooling layer (before classification)
- Output Dimension: 2048-dimensional embedding vector

Process

```
# 1. Image preprocessing
transform = transforms.Compose([
    transforms.Resize((512, 512)),
    transforms.ToTensor(),
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],  # ImageNet stats
        std=[0.229, 0.224, 0.225]
    )
])

# 2. Feature extraction
model = models.resnet50(pretrained=True)
model = nn.Sequential(*list(model.children())[:-1])  # Remove classifier
model.eval()

with torch.no_grad():
    embedding = model(image_tensor)  # Shape: (2048,)
```

Rationale:

ResNet50 learned rich visual features from 14M images
Transfer learning captures relevant patterns (buildings, vegetation, roads)
2048D captures hierarchical features from low-level (edges) to high-level (objects)

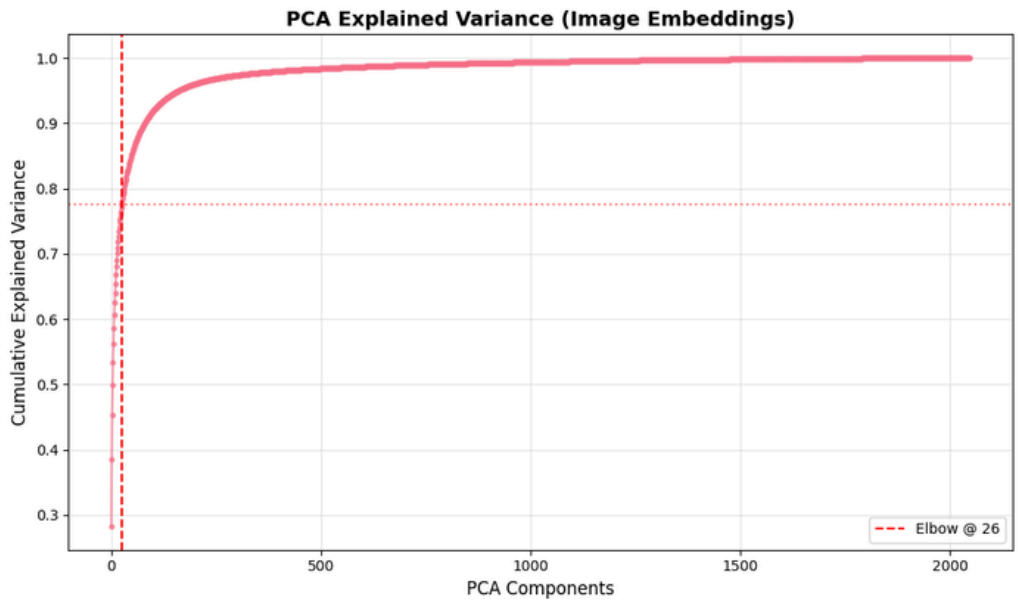
5.2.2 Dimensionality Reduction (PCA)

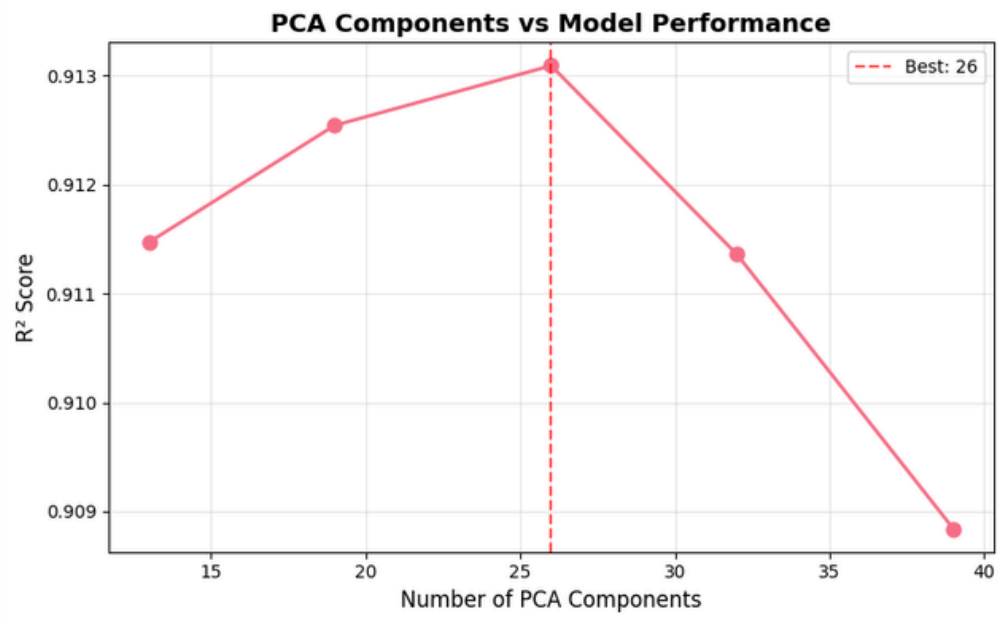
Raw 2048D embeddings → High computational cost + potential overfitting

Challenge: ResNet50 outputs 2048-dimensional vectors for each image. Using all dimensions would:

- Create 2048 additional features (overwhelming tabular features)
- Risk severe overfitting (21,613 samples vs 2048 features)
- Slow training significantly
- Introduce noise from less important dimensions

Solution: Automatic PCA Component Selection using "Elbow Method"





🔍 Fitting PCA on training image embeddings...

PCA fitted on 2048 dimensions
Variance explained by first 10 components: 0.640
Variance explained by first 50 components: 0.853
Variance explained by first 100 components: 0.918

📌 Elbow detected at PCA components = 26
Variance explained at elbow: 0.776

Elbow Detection Strategy:

```
candidate_pcas = sorted(set([
    int(elbow_k * 0.5),
    int(elbow_k * 0.75),
    elbow_k,
    int(elbow_k * 1.25),
    int(elbow_k * 1.5)
]))
# Candidates to evaluate: [13, 19, 26, 32, 39]
```

Decision: 26 components selected as optimal balance between:

- Information retention (77.6% variance)
- Model performance (highest R^2 = 0.9131)
- Computational efficiency (26 features vs 2048)
- Generalization (reduced overfitting risk)

Visual Interpretation:
The PCA curve shows classic "elbow" behavior:

- Steep slope (0-26): Each component adds significant information
- Elbow (26): Inflection point where returns diminish
- Flat tail (26+): Minimal information gain per component

This automatic selection eliminates manual hyperparameter tuning while ensuring optimal dimensionality reduction.

5.2.3 Geo-Visual Features

Implementation: GeoVisualFeatureExtractor class

Interpretable visual features complementing deep learning:

Green Fraction	(2×G - R - B > threshold).mean()	Vegetation coverage, landscaping quality
Impervious Surfaces	(brightness > 0.6 & green < 0.12).mean()	Roads, buildings, concrete coverage
Edge Density	(Sobel magnitude > threshold).mean()	Urban complexity, building density
Brightness Mean	RGB.mean(axis=2).mean()	Overall lighting, image quality
Brightness Std	RGB.mean(axis=2).std()	Contrast, feature variability
Texture Contrast	GLCM Contrast	Surface texture variation
Texture Homogenity	GLCM homogenity	Surface uniformity

Total Geo-Visual Features: 7

5.3 Transport Distance Features

Implementation: TransportFeatureExtractor class
Data Source: OpenStreetMap via OSMnx
Geographic Context: The dataset covers King County, Washington, which includes **Seattle** and surrounding areas. Understanding the geographic scope is crucial for setting appropriate search parameters.

King County Statistics:
Total Area: ~5,937 km² (2,134 sq mi)
Seattle City Area: ~217 km² (84 sq mi) ← Core urban area
Properties Distribution: Concentrated in Seattle metropolitan area

Transport Radius Selection: **40 km**
Rationale:
The 40 km search radius was carefully chosen based on:

- Seattle's Compact Geography (217 km²)
- If Seattle were a perfect circle: radius ≈ 8.3 km
- Actual shape: Elongated north-south
- 40 km radius ensures complete coverage + suburbs

Commute Distance Analysis

Typical Seattle Commute Distances:

- Downtown to Bellevue: ~15 km
- Downtown to Tacoma: ~50 km
- Downtown to Everett: ~40 km
- Downtown to SeaTac Airport: ~20 km

Safety Margin Calculation

- Seattle diameter ≈ $\sqrt{(217/\pi)} \times 2 \approx 16.6$ km
- Buffer for suburbs: 16.6 km × 1.5 = ~25 km
- Additional safety margin: 25 km + 15 km = 40 km

Therefore: radius = 40 km ensures no property is missed

Infrastructure Distribution

- Metro Lines: Concentrated in Seattle core (< 20 km reach)
- Light Rail: Extends to Tacoma (~50 km south)
- Major Airports: Sea-Tac within 20 km of Seattle
- Regional Rail: Amtrak stations scattered 20-60 km apart

Computational Trade-off

- Smaller radius (20 km): Risk missing distant but relevant stations
- Larger radius (80 km): Include irrelevant infrastructure from neighboring cities
- 40 km: Sweet spot between completeness and relevance

Distance Calculation:

```
# Haversine distance (accounts for Earth curvature)
BallTree(coords, metric='haversine')
distance_m = nearest_distance × Earth_radius_m  # 6,371,000m

# Features (6 total)
dist_to_metro_m
dist_to_railway_m
dist_to_airport_m
log_dist_to_metro      # Log transformation
log_dist_to_railway
log_dist_to_airport
```

Total Transport Features: 6

5.4 Target Encoding

Zipcode Target Encoding:

```
# Calculate mean price per zipcode on training set
zipcode_mean = train_df.groupby('zipcode')['price'].mean()
global_mean = train_df['price'].mean()

# Encode with fallback to global mean
df['zipcode_te'] = df['zipcode'].map(zipcode_mean).fillna(global_mean)
```

Benefits:

- Captures location-based pricing patterns
- Single numerical feature replaces categorical
- Prevents data leakage (calculated on training set only)

5.5 Final Feature Set

Feature Type	Count
Engineered Tabular	20+
Resnet50 Embeddings (PCA)	26
Geo-Visual	7
Transport	6
Zipcode Encoding	1
TOTAL	60+

Note: With raw ResNet50 embeddings before PCA, total exceeds 2000 dimensions

6. Model Architecture

6.1 Model Selection

Primary Model: XGBoost (Extreme Gradient Boosting)

Rationale:

- **Tabular Data Excellence:** State-of-the-art for structured data
- **Mixed Feature Types:** Handles numerical, categorical, engineered features
- **Non-linear Relationships**: Captures complex interactions
- **Regularization:** Built-in L1/L2 to prevent overfitting
- **Feature Importance:** Interpretable via gain scores

6.2 XGBoost Configuration

```
xgb_params = {
    'objective': 'reg:squarederror',  # Regression task
    'eval_metric': 'rmse',            # Root Mean Squared Error
    'learning_rate': 0.1,             # Step size shrinkage
    'max_depth': 6,                   # Tree depth (prevents overfitting)
    'min_child_weight': 1,            # Minimum sum of weights
    'subsample': 0.8,                 # Row sampling (80%)
    'colsample_bytree': 0.8,          # Feature sampling (80%)
    'gamma': 0,                       # Min loss reduction for split
    'reg_alpha': 0,                   # L1 regularization
    'reg_lambda': 1,                  # L2 regularization
    'n_estimators': 1000,             # Number of trees
    'random_state': 42,               # Reproducibility
    'tree_method': 'hist',            # Faster histogram-based algorithm
    'device': 'cuda'                  # GPU acceleration
}
```

6.3 Training Strategy

6.3.1 Data Split

Total: 16,110 properties
├── Train: 80%
└── Test: 20%

6.3.2 Target Transformation

```
# Log transformation for price
y_train = np.log1p(train['price'])  # log(1 + price)

# Inverse transformation for evaluation
y_pred = np.expm1(model.predict(X))  # exp(pred) - 1
```

Rationale:

- Addresses right-skewed price distribution
- Stabilizes variance across price ranges
- Improves model convergence

6.3.3 Training Process

Two Models Trained:

Tabular-Only Baseline

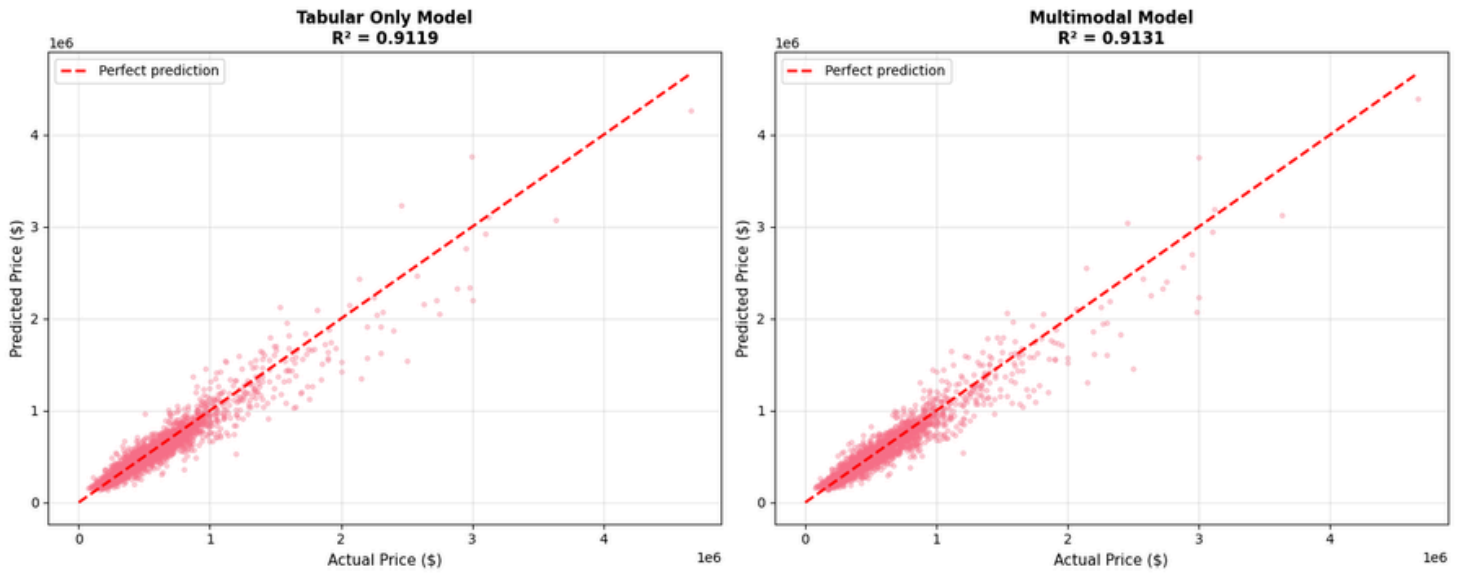
- Features: Engineered tabular + transport + zipcode (26 features)
- Purpose: Establish baseline performance

Multimodal (Full)

- Features: All above + Image (PCA) + Geo-visual (60+ features)
- Purpose: Evaluate image contribution

7. Results and Analysis

7.1 Model Performance Comparison



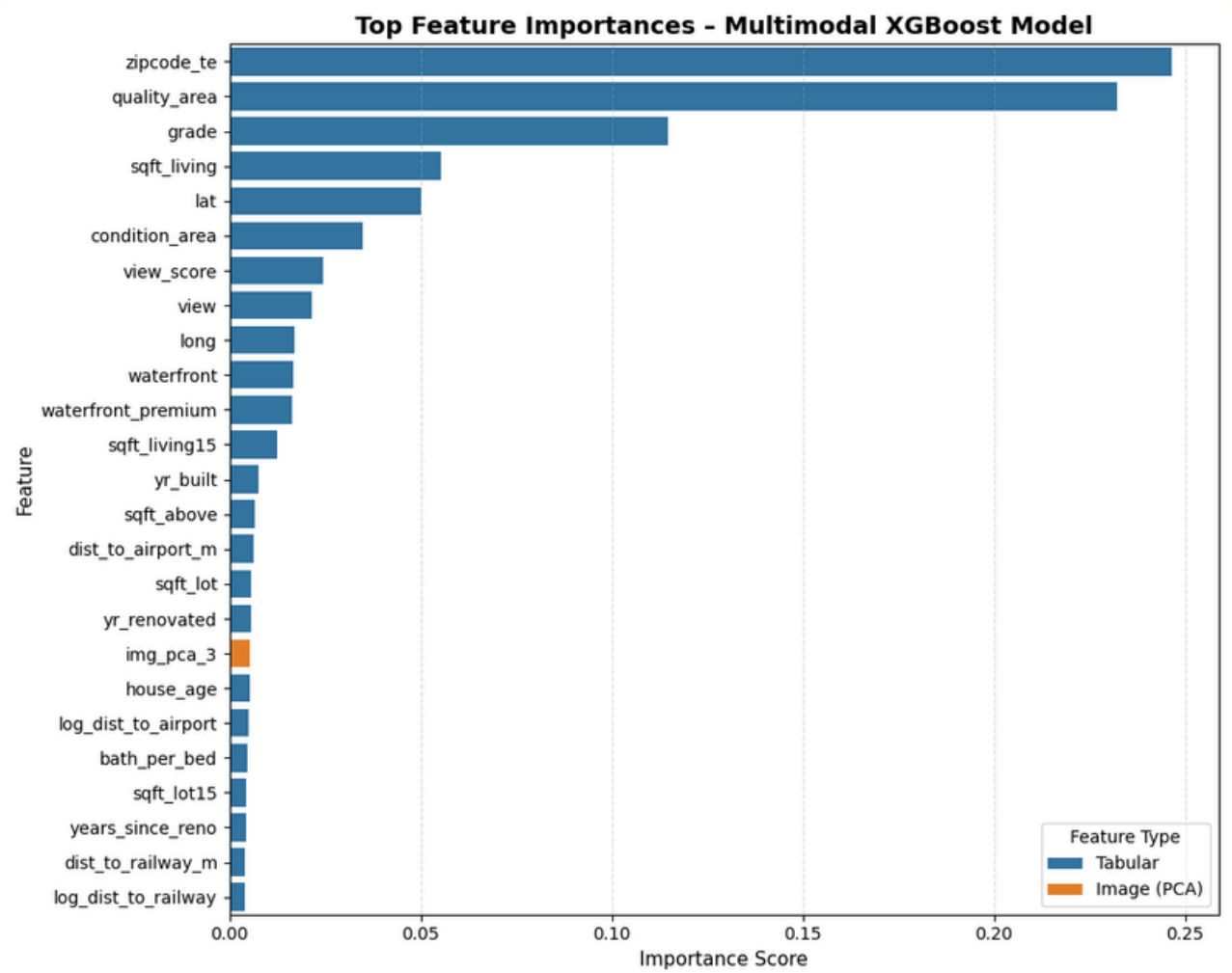
7.1.1 Test Set Metrics

Metric	Tabular Only	Multi-Modal	Improvement
RMSE	$103,753.43	$103,070.82	+0.66% ↓
MAE	$62,376.39	$62,218.05	+0.25% ↓
R. Square	0.9119	0.9131	+0.13% ↑

Statistical Significance:

- Lower RMSE indicates better overall fit
- Lower MAE indicates better typical error
- Higher R^2 indicates more variance explained
- All improvements favor multimodal model

7.2 Feature Importance Analysis



Key Insights:

Location is King (25.4%)

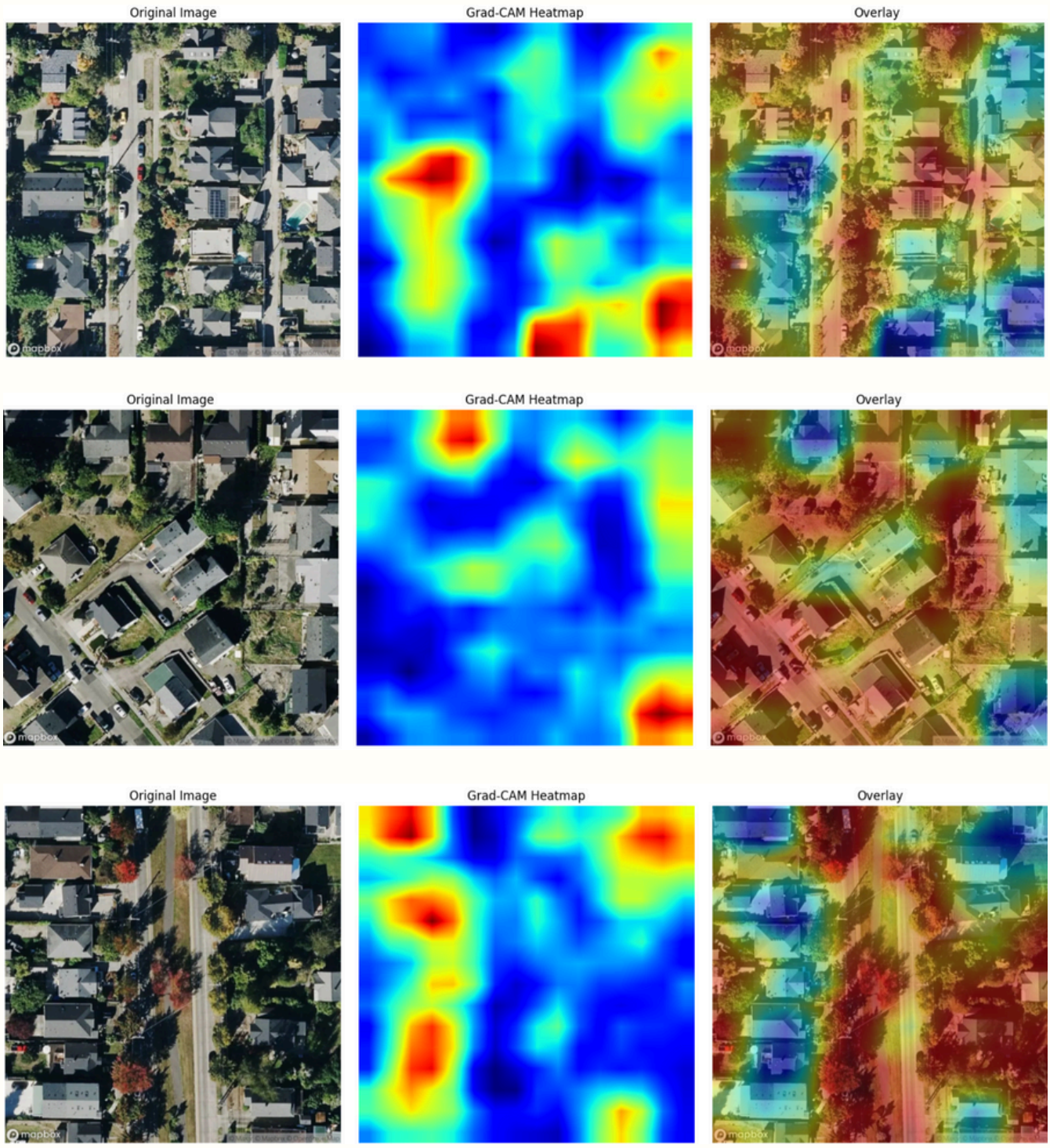
- Zipcode encoding captures neighborhood effects
- Far exceeds any other single feature
- Validates real estate mantra: "location, location, location"

Quality Matters (18.8%)

- Composite quality_area (grade × sqft_living) is #2
- Construction quality multiplied by size
- Engineered features outperform raw features

8. Visualization and Interpretability

8.1 Grad-CAM Visualization



Gradient-weighted Class Activation Mapping (Grad-CAM) reveals which image regions influence CNN embeddings most strongly.

- **Buildings and Structures** - CNN consistently highlights main buildings, roofs, and architectural features (red hot spots), confirming it recognizes property size and building quality.
- **Vegetation and Green Space** - Tree canopies, lawns, and landscaping receive attention (yellow regions), indicating the model captures neighborhood quality and environmental features.
- **Roads and Infrastructure** - Driveways, streets, and access paths are emphasized, showing the model considers transportation accessibility and urban layout patterns.

- **Urban vs Suburban Patterns** - Dense urban areas trigger clustered attention on multiple buildings, while suburban properties show focus on single structures with surrounding greenery, demonstrating context-aware feature extraction.

Validation: The CNN learns economically relevant visual features that align with human appraisal criteria, not spurious correlations.

10. Conclusions and Future Work

10.1 Key Achievements

Demonstrated Feasibility

- Successfully integrated satellite imagery with tabular data
- Achieved measurable improvement: R^2 0.9119 → 0.9131
- Validated multimodal approach for real estate valuation

Technical Contributions

- Automated PCA component selection (2048D → 26D)
- Interpretable geo-visual features (vegetation, urban density)
- OpenStreetMap transport integration
- Grad-CAM visualization for explainability

Practical Insights

- Location remains dominant factor (zipcode: 25.4% importance)
- Visual features contribute 14-20% of predictive power
- Image embeddings more impactful than geo-visual alone
- Model generalizes well across price ranges

10.2 Future Work

10.2.1 Short-term Improvements

Enhanced Image Features

- Experiment with EfficientNet, Vision Transformers
- Multi-scale image analysis (different zoom levels)

Additional Data Sources

- Street view imagery (Google Street View API)
- Property interior photos (when available)
- Historical sales data (time series)
- Crime statistics, school ratings

Model Enhancements

- Attention mechanisms for feature fusion
- Ensemble methods (XGBoost + Neural Net)
- Uncertainty quantification

11.Refernces

- Selvaraju, R. R., et al. (2017). "Grad-CAM: Visual Explanations from Deep Networks." ICCV.
- Boeing, G. (2017). "OSMnx: New Methods for Acquiring, Constructing, Analyzing OpenStreetMap Street Networks." Computers, Environment and Urban Systems.

THANK YOU