Zillow Home Value Prediction Using Machine Learning and Big Data Techniques

Scalable Modeling of California Property Prices with PySpark and XGBoost

Harshith Makkapati
Department of Computer Science
University of Houston
Houston, USA

Srilekha Rayedi
Department of Computer Science
University of Houston
Houston, USA

Abstract— **We aim to forecast house prices precisely with advanced machine learning techniques and large-scale processing of the Zillow Prize 1 data. We trained on a set of around 3 million Californian homes, meticulously cleaned data, and created features before tuning machines like XGBoost and Random Forest. Our best model achieved an RMSE of $270,100 and an R^2 of 0.54, indicating moderately strong predictive performance on unseen data. We describe the key data pre-processing tasks, model building techniques, and future research in real estate forecasting.**

Keywords— Home value prediction, XGBoost, Random Forest, PySpark, Zillow, real estate analytics, big data

Introduction

Real estate appraisal is at the core of buying or selling, investing, and policymaking. The majority of traditional approaches are based on historical patterns, valuation heuristics, and subject-objective expert judgment that are not scalable to contemporary diversified housing markets. With more and more information about properties, machine learning gives us a very effective tool kit for pattern recognition, large-scale data handling, and building improved and more efficient predictive models compared to traditional techniques.

Zillow, the leading U.S. real estate online platform, initially created the Zestimate—a computerized valuation system that ran millions of homes through artificial intelligence. In an effort to further advance the ability of computer real estate appraisal, Zillow launched the Zillow Prize competition that invited data scientists to enhance its predictive power using the current dataset and external data. Our project approaches the challenge head-on with PySpark to upscale preprocessing and popular machine learning algorithms such as XGBoost and Random Forest to execute regression modeling. Implementing judicious feature engineering, outlier rejection, and hyperparameter tuning, we uncover a strong pipeline that demonstrates the capabilities of solution-driven data solutions to real estate price forecasting.

I. Dataset Overview

A. Data Preparation

We used the Zillow Prize 1 dataset from Kaggle, which includes ~2.9 million entries with 58 features. These cover property details (bedrooms, bathrooms, square footage), geographic markers (latitude, longitude, regionidzip), and financials (tax amount, value).

Feature Selection: We retained only statistically and domain-relevant features. Discarded features had excessive missingness or low relevance.

B. Cleaning and Preprocessing

- Missing numeric values (e.g., bathroomcnt, bedroomcnt) were imputed using the median.
- Categorical variables were filled with "Unknown".
- Outliers in features like lotsizesquarefeet were filtered using thresholds.
- Data was processed with PySpark for scalable analysis.

II. Exploratory Data Analysis

We conducted thorough exploratory data analysis (EDA) to determine the shape and distribution of important features in the data and guide feature selection and transformation throughout the analysis.Bedroom and Bathroom Count: Most houses were 2–4 bedrooms and 2–3 bathrooms. Both variables were highly right-skewed with hardly any luxury houses having more than 6 bedrooms or 5 bathrooms.

Square Footage and Lot Size: Mean computed finishedsquarefeet variable was 1,000 to 2,500 sq ft, and lotsizesquarefeet had very right skew with outliers (i.e., lots of more than 1 million sq ft), which were removed after outliers removal.
Tax and Structure Value: taxvaluedollarcnt, structuretaxvaluedollarcnt, and landtaxvaluedollarcnt were all right-skewed with lengthy tails such as real estate price distributions would have had. The distributions showed most homes to be of low value but that a few expensive homes weighted the scale.

House Age Distribution: Up to 2016 - yearbuilt, most of the houses are 30–60 years old. We see that most of the building took place in the 1960s and 1970s. There are fewer houses which are recent (newer than 2010) or old (older than 1940).

Garage and Pool Presence: garagecarcnt varied from 0 to 2, the most common of which was the 2-car garage. poolcnt was skewed to extremes—in houses there weren't pools, but there were pools sometimes. Pool presence would therefore be a differential luxury feature.

Correlation Analysis: Fig. 1 heat map revealed significant linear relationships: number of rooms and number of bedrooms (~0.73), number of bathrooms and square feet (~0.81), square feet and taxvaluedollarcnt (0.52). Negative linear relationship between age of house and property value (-0.31) was evidence of depreciation effect.

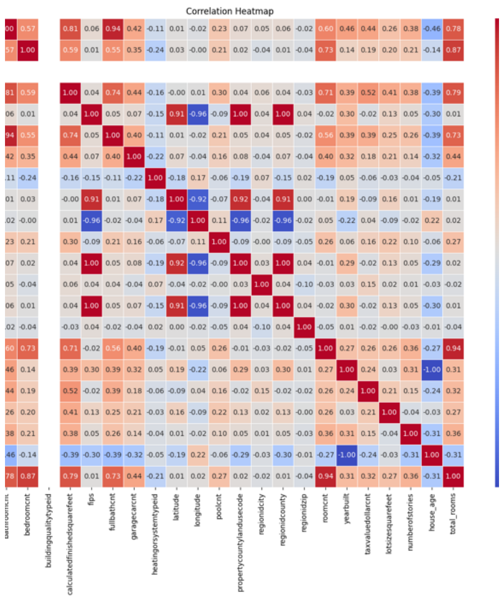


Fig. 1 – Correlation heatmap of numerical features

Geospatial Visualization: California's zipcode map showed the geographical distribution of properties in major metropolitan areas like Los Angeles and Orange County (Fig. 3). K-Means clustering (Fig. 4) also used to divide geographical space into latitude and longitude and identify clusters of adjacent properties that may have similar patterns of price or development.

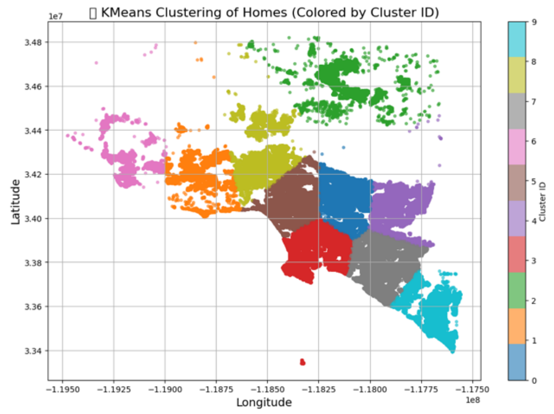


Fig. 2 – K-Means clustering of properties by location

III. Feature Engineering

We required feature engineering to achieve improved prediction accuracy and model explainability. We utilized domain-knowledge-based features and statistical manipulations to extract signal information from raw data.

House Age: Our new feature house_age we calculated as a house_age = 2016 - yearbuilt
The feature would help in visualizing depreciation patterns as well as maintenance patterns generally comparable with market value.

Area per Room: We estimated average area per room from finishedsquarefeet/roomcnt. It is an indicator of space efficiency and can differentiate between denseliberally spaced and open house types.

Value Ratrices: Indicators such as structure_to_land_ratio, structuretaxvaluedollarcnt to landtaxvaluedollarcnt, were used in attempts to estimate the building value to land value ratio within a property. Synthetic ones such as these help to estimate the value composition that is convenient when in heterogeneous markets.

Log Transformations: We employed logarithmic scaling to attempt to accommodate heavy-tailed distributions in values like taxvaluedollarcnt. It improves the ability of the model to learn from values of different orders of magnitude.

Label Encoding: For categorical features like propertycountylandusecode, we employed label encoding to avoid compromising the identification of names when inducing sparsity. We can then feed these into models like XGBoost without needing to create high-dimensional one-hot encodings.

Geographic Clustering: We employed K-Means clustering to group houses by latitude and longitude and have derived a new geo_cluster feature. This worked well for us in clustering properties into place segments, respecting spatial effects impossible for discrete points.

These features were tested thoroughly with correlation analysis and exploratory plots to ensure that they improved downstream model learning and interpretability.
Computed area per room for better insight.
Label encoding is used for categorical variables like landusecode.

IV. Model Architecture

The architecture of our home value prediction system is designed to be modular, scalable, and easily adaptable for future enhancements. It is organized into three major layers: **Data Layer**, **Modeling Layer**, and **Application Layer**, each serving a distinct role in the pipeline.

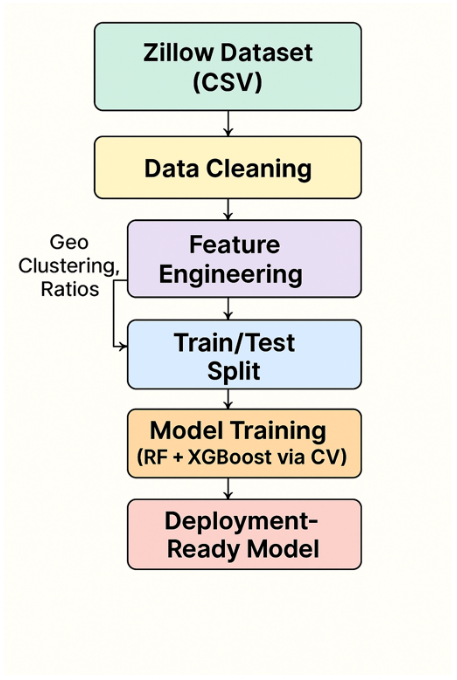


Fig. 3 – Model architecture pipeline

A. Data Layer (Preprocessing and Feature Engineering)

This layer is responsible for **ingesting**, **cleaning**, and **transforming** raw real estate data from the Zillow Prize 1 dataset.|
Tools: PySpark, Pandas
Missing Value Handling: Median imputation for numeric fields; "Unknown" for categorical variables.
Outlier Removal: Applied threshold filtering to extreme values (e.g., lotsizesquarefeet > 1M).
Log Transformation: Applied to skewed financial features like taxvaluedollarcnt
Categorical Encoding: Used label encoding on variables like landusecode.
Geo Clustering: Applied K-Means on lat/long to generate a geo_cluster feature.
The output of this layer is a **cleaned, feature-rich PySpark DataFrame** suitable for scalable model training.

B. Modeling Layer

This layer includes the core **machine learning models** trained to predict property values based on engineered features.

1. Model Selection

We experimented with multiple models, but two ensemble methods were selected for comparison:

- **Random Forest Regressor** (baseline)
- **XGBoost Regressor** (final model)

2. Training Process

- **Train/Test Split**: 70/30 split with stratified sampling on key features (e.g., home value range).
- **Hyperparameter Tuning**: Performed using RandomizedSearchCV for XGBoost.
- **Target Variable**: taxvaluedollarcnt (log-transformed for normality)

4. Evaluation Metrics

- **MAE (Mean Absolute Error)**: $156,820.25
- **RMSE (Root Mean Squared Error)**: $270,100.75
- **R² Score**: 0.5415

These results indicated strong performance on large-scale tabular data.

C. Application Layer (Deployment-Ready Output)

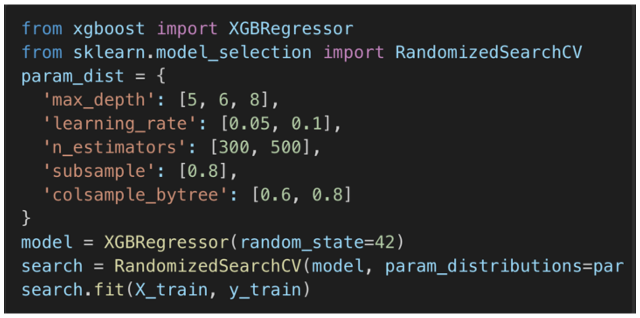
While full deployment was outside the project scope, the architecture is designed to support future deployment with minimal effort.

- **Scalable Data Pipeline**: All preprocessing and feature engineering steps are reusable in a production PySpark pipeline.
- **Model Serialization**: Final XGBoost model saved using joblib, ready for use in Flask or Streamlit apps.
- **Input Compatibility**: Model accepts structured input including:
  Bedrooms, bathrooms, square footage
  Year built, zip code, geo_cluster, and more
- **Extendable APIs**: Could easily be wrapped into a RESTful API for inference.

IV. Model Training and Evaluation

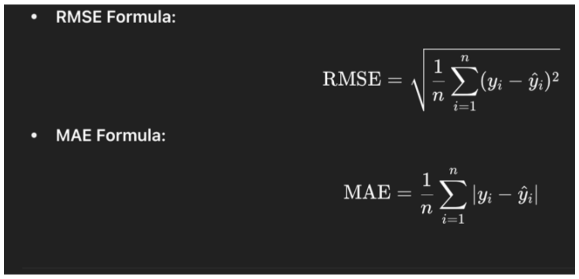
A. XGBoost Chosen for its efficiency on tabular data, missing value handling, and regularization. Best parameters (via RandomizedSearchCV):

- learning_rate = 0.05
- n_estimators = 500
- max_depth = 6
- subsample = 0.8
- colsample_bytree = 0.6 Achieved:
- MAE: $156,820.25
- RMSE: $270,100.75
- R²: 0.5415



```
from xgboost import XGBRegressor
from sklearn.model_selection import RandomizedSearchCV
param_dist = {
  'max_depth': [5, 6, 8],
  'learning_rate': [0.05, 0.1],
  'n_estimators': [300, 500],
  'subsample': [0.8],
  'colsample_bytree': [0.6, 0.8]
}
model = XGBRegressor(random_state=42)
search = RandomizedSearchCV(model, param_distributions=par
search.fit(X_train, y_train)
```

Fig. 4 – XGBoost performance metrics



- RMSE Formula:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

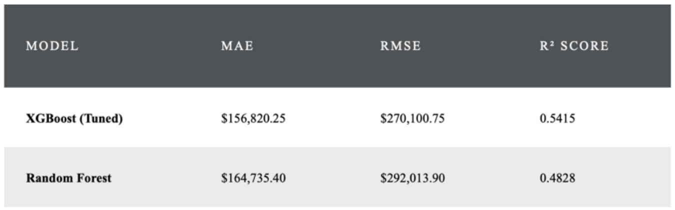
- MAE Formula:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Fig. RMSE and MAE Formulae

B. Random Forest Used as baseline comparison:

- n_estimators = 100
- max_depth = None Performance was slightly lower than XGBoost.

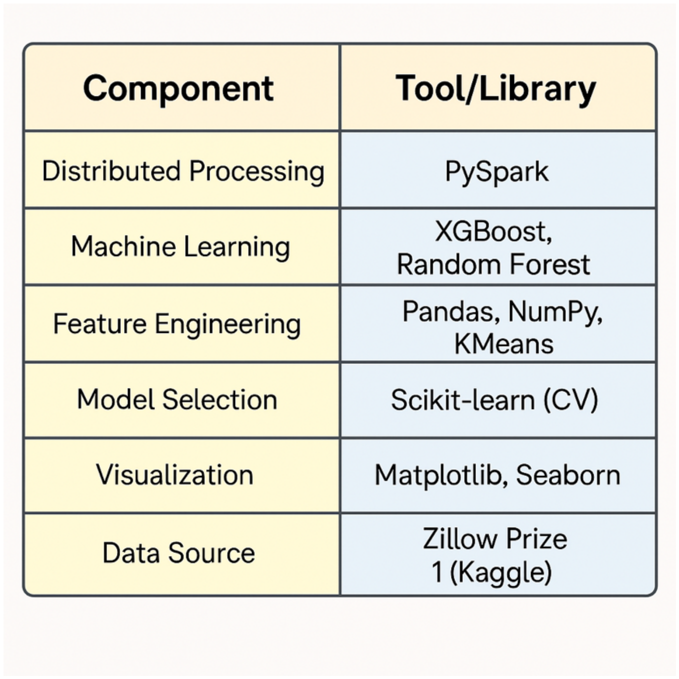


MODEL	MAE	RMSE	R² SCORE
XGBoost (Tuned)	$156,820.25	$270,100.75	0.5415
Random Forest	$164,735.40	$292,013.90	0.4828

Fig. 5 – Random Forest vs XGBoost comparison

To forecast home values, we tested two machine learning models: Random Forest and XGBoost (tuned). In every metric, the adjusted XGBoost model performed better than Random Forest:
With a lower RMSE of $270,100.75 and a lower MAE of $156,820.25, XGBoost demonstrated superior management of greater errors and smaller average prediction errors, respectively.
Additionally, it produced a higher R2 score of 0.5415, which indicates that it accounted for around 54% of the variation in home values.
Despite being a strong baseline, Random Forest had a lower R2 score (0.4828), a larger MAE ($164,735.40), and a higher RMSE ($292,013.90).
These outcomes demonstrate that, for this regression challenge, XGBoost was the more accurate and dependable model.

Tech stack used:



Component	Tool/Library
Distributed Processing	PySpark
Machine Learning	XGBoost, Random Forest
Feature Engineering	Pandas, NumPy, KMeans
Model Selection	Scikit-learn (CV)
Visualization	Matplotlib, Seaborn
Data Source	Zillow Prize 1 (Kaggle)

V. Challenges

We had a number of significant obstacles during the project that needed to be handled carefully. Managing missing data in crucial features like garagecarcnt, poolcnt, and numberofstories—where missing values were common and not random—was one of the main challenges. Carefully choosing imputation techniques was necessary to maintain data integrity without adding bias. Eliminating extreme outliers presented another difficulty, particularly for variables such as lotsizesquarefeet, which showed unusually high values that could skew model training and inflate error metrics. In order to ensure that the model captured significant patterns without learning noise in the training data, we also had to balance model complexity to prevent overfitting, especially while adjusting hyperparameters in XGBoost. Building a dependable and broadly applicable prediction pipeline required addressing these issues.

VI. Conclusion

In conclusion, our project demonstrates the value of combining domain knowledge with scalable big data tools and machine learning to improve home value prediction. Our pipeline—from PySpark-based preprocessing to fine-tuned XGBoost modeling—offers a robust foundation for real-world real estate analytics.

VII. Future Work

Future research might include integrating temporal data like the month and year of sale to capture seasonal trends and real estate market cycles, which would further improve the accuracy and resilience of our home value prediction model. furthermore, adding external datasets—such as crime statistics at the neighborhood level, school district rankings, and economic indicators like employment rates or median income—could provide insightful background information that affects property value in ways other than structural factors alone. experimenting with deep learning models designed for tabular data, such tabnet or transformer-based architectures, which have demonstrated the capacity to model intricate, non-linear relationships while requiring less manual feature engineering, is another exciting avenue. these improvements have the potential to greatly increase forecast accuracy and provide more profound understanding of market dynamics.

VIII. Background and Related Work

Traditional real estate valuation methods rely on expert heuristics or linear regressions, but they often fail in capturing nonlinear patterns and large-scale trends. Recent research leverages ensemble models like XGBoost and Random Forest for superior accuracy. Our project builds on these approaches while integrating distributed processing (PySpark) and geospatial clustering to scale across millions of records and improve predictive power.

IX. Limitations

Our model has some limitations even if it shows good prediction accuracy. First, the model may not be able to generalize to housing markets in other states with distinct economic, geographic, and regulatory variables because the information is restricted to properties in California. Second, temporal features that could capture seasonality and market swings over time, including the month or year of sale, are not currently included in our research. Last but not least, the dataset contains a number of weakly populated characteristics, such poolcnt and garagecarcnt, which may hinder the model's capacity to properly understand their influence on home values. Future iterations may increase generality and model accuracy by addressing these restrictions.

X. Acknowledgement

We would like to express our heartfelt gratitude to **Feng Yan** for their continuous support, encouragement, and insightful guidance throughout the course of this project. Their expertise and feedback were instrumental in shaping our understanding of machine learning techniques and their application to real-world datasets. We are truly thankful for the opportunity to work under their mentorship, which significantly enhanced the quality and depth of our work.
We also thank **Zillow Group** and **Kaggle** for providing the dataset and hosting the Zillow Prize competition, and we appreciate the collaborative efforts of our peers and colleagues who provided valuable suggestions during the development of this project.

XI. References

[1] Hjort, A., Pensar, J., Scheel, I., & Sommervoll, D. E. (2022). House price prediction with gradient boosted trees under different loss functions. *Journal of Property Research*, *39*(4), 338–364.

[2] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, San Francisco, CA, USA, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[3] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[4] Apache Spark, *PySpark Documentation*, Apache Software Foundation. Accessed: Apr. 28, 2025.

[5] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[6] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: With Applications in R*, 1st ed. New York, NY, USA: Springer, 2013.

[7] OpenCage Geocoder, "Worldwide ZIP Code Data." Accessed: Apr. 28, 2025.

[8] U.S. Census Bureau, "ZIP Code Tabulation Areas (ZCTAs)." Accessed: Apr. 28, 2025.

[9] Zillow Prize: Zillow's Home Value Prediction (Zestimate), Kaggle. Accessed: Apr. 28, 2025.