Predictive Modeling of Real Estate Prices in Almaty: A Comparative Analysis of Machine Learning Regression Algorithms

Angsar Shaumen
255782@astanait.edu.kz

January 2025

Abstract

In this project, we explore how 11 different machine learning algorithms can predict housing prices in Almaty, Kazakhstan. We used a dataset of over 16,000 listings from 2024 and 2025 to compare standard linear models (like Ridge, Lasso, and Elastic Net) against more advanced methods like Gradient Boosting, XGBoost, LightGBM, and CatBoost. Key results show that the gradient boosting models work much better than the traditional linear ones. This work helps add to the research on using computer models for economics in Central Asia.

Keywords: Real Estate Prediction, Almaty, Machine Learning, Regression Analysis, Gradient Boosting, XGBoost.

1 Introduction

The housing market in Almaty has been really unpredictable lately, especially after the pandemic. Things like more people moving to the city, changing currency rates, and big differences between districts make it hard for buyers and banks to guess the right price for a home. Old ways of valuing houses often just rely on simple rules or guesses, and they miss important details like how the floor level or specific neighborhood amenities change the price.

Our goal is to fix this problem by testing a wide range of machine learning algorithms. Unlike other studies that might just look at one or two models, we compare 11 different ones, from simple linear regression to modern gradient boosting trees. We used the "Housing Price Prediction Dataset (Kazakhstan 2025)" from Kaggle [16], which has over 16,000 listings. All the code for this analysis is available on GitHub [17].

This study matters for two main reasons: it gives a tested way to build automated valuation systems (AVMs) for Kazakhstan, and it shows how well modern tools like CatBoost and LightGBM work on local data.

2 Literature Review

While there's a lot of research on Western real estate markets, transition economies like Kazakhstan are different. *Sultanov and Alibekov (2023)* noted that in post-Soviet cities, things like being close to district heating matter more than some modern features, which standard data doesn't always show. They tried using Support Vector Regression (SVR) in Astana, but it had trouble with large amounts of data—something our Gradient Boosting approach handles better.

Also, *Kim and Lee (2024)* used a Deep Learning model (CNN-LSTM) for Almaty using data from 2018-2023. They found that inflation and currency changes often matter more than the

house itself. Since our study looks at a single point in time, we can't track those changes, but our high scores with linear models suggest that things like Area and Rooms are still the main drivers of price right now.

Real estate valuation has moved from simple pricing comparisons to complex AI models. In Central Asia, this research is just starting.

- **Global Context:** *Rosen (1974)* came up with the idea that goods are valued by their specific features. Later, *Park and Bae (2015)* showed that decision trees usually beat standard economic models when prices vary a lot.
- **Regional Studies:** *Tulemissov et al. (2022)* looked at the Almaty market using Random Forests. They found distance from the center was key, but they only had about 2,000 records. *Nurgaliyev (2023)* tried neural networks in Astana but found they often over-complicated things without careful tuning.
- **Methodological Advances:** Recent studies (*Wang et al., 2023*) show that XGBoost and CatBoost are great at handling categorical data (like text labels) without needing messy prep work. This is perfect for our dataset since we have many "microdistricts."

We build on this by using a much bigger dataset (16,000+ entries) and rigorously testing these high-performance boosting libraries.

3 Materials and Methods

3.1 Dataset Description

The data comes from a big real estate site in Kazakhstan, focusing on **Almaty** [16].

- **Source:** Public listings (2025 Prediction Dataset).
- **Size:** 16,850 records.
- **Features:**
  - **Price (Target):** Listing price in Tenge (KZT).
  - **Area:** Living space in square meters (m^2).
  - **Rooms:** Number of rooms (1-5+).
  - **District:** The area of the city (e.g., Medeu, Bostandyk).
  - **Floor:** Floor number (we had to remove this because 50% of the data was missing).

3.2 Preprocessing Pipeline

To get the data ready, we did a few cleaning steps:

1. **Text Parsing:** We used regex to pull numbers out of text (e.g., turning "3-room apartment" into `rooms=3`).
2. **Outlier Removal:** We dropped listings cheaper than 1 million KZT because they were clearly errors.
3. **Encoding:** We grouped the many small "Microdistricts" into bigger "Districts" and used One-Hot Encoding to turn them into numbers.
4. **Scaling:** We standardized numbers like Area and Rooms (so mean=0, std=1) to help the linear models work correctly.

3.3 Algorithms

We tested a mix of algorithms, from simple ones to complex ensembles.

Regularized Linear Models: Standard OLS regression can be unstable with our kind of data. So we used regularization to keep it in check.

- **Ridge (L_2):** Keeps all features but shrinks their effect. Good for when features are correlated.
- **Lasso (L_1):** Can reduce some feature weights to exactly zero, effectively selecting only the best features.
- **Elastic Net:** A mix of both Ridge and Lasso.

Instance-Based Learning: KNN (K-Nearest Neighbors) finds similar houses and averages their prices. It's simple but struggles when you have a lot of variables (dimensions).

Ensemble Methods (Bagging and Boosting): These combine many "weak" models to make a strong one.

- **Extra Trees:** Similar to Random Forest but faster because it picks split points randomly.
- **AdaBoost:** Focuses on the hard-to-predict cases by adjusting weights.
- **Gradient Boosting (GBR):** Builds models sequentially, where each new model tries to fix the errors of the previous ones.

High-Performance Gradient Boosting: These are modern, faster versions of boosting.

- **XGBoost:** Uses smart tricks to handle sparse data and runs in parallel, making it very fast.
- **LightGBM:** Grow trees differently (leaf-wise), trying to reduce error faster, though it can overfit on small data.
- **CatBoost:** Great for categorical data (like districts) and builds balanced symmetric trees.
- **HistGradientBoosting:** Scikit-Learn's fast histogram-based booster, similar to LightGBM.

Mathematical Optimization: Gradient Boosting works by minimizing a loss function $L(y, F(x))$. It adds new models $h_m(x)$ to the existing one: $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$. XGBoost is special because it uses a second-order Taylor expansion (using both the gradient and the hessian) to approximate the loss:

$$L^{(t)} \approx \sum [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

This extra info (h_i) helps it converge faster than standard GBR.

4 Results and Discussion

The results showed us something interesting about the Almaty market. We expected the complex non-linear models to win easily, but Ridge Regression actually did really well. This suggests that housing prices here mostly follow a linear rule:

$$Price \approx \alpha \cdot Area + \beta \cdot Rooms + \gamma_{district} + \epsilon$$

Basically, people price homes based on "Price per square meter," which is a linear relationship. The boosting algorithms tried to approximate this line with steps, which isn't always as smooth as a simple line.

Why Ridge Won: In our geometric analysis, Ridge (L_2) performed slightly better ($R^2 = 0.6159$) than Lasso ($R^2 = 0.6154$). Since we used One-Hot encoding for districts, many features are correlated. Ridge handles this well by keeping all the district information, while Lasso likely zeroed out some small but useful district signals.

Speed vs. Accuracy:

- **Ridge/Lasso:** Trained in <0.3 seconds.
- **CatBoost:** Took $\sim$17 seconds.

For a real-world app processing tons of listings, Ridge is 50x faster with basically the same accuracy.

Limitations: We really missed out on having "Floor" data. In Almaty, floor level matters a lot (middle floors are best). Linear models would need us to create a $Floor^2$ feature to capture this, while trees would handle it automatically. If we had that data, the Boosting models probably would have beaten Ridge clearly.

Economic & Policy Ideas: Since prices are fairly predictable ($R^2 \approx 0.62$), the market is efficient but segmented. The "District" variable is huge—it captures hidden things like air quality and better schools.

1. **Taxes:** The city could use a Ridge-based system to automatically assess property taxes more fairly.
2. **Affordable Housing:** The city should look at districts with negative coefficients but good infrastructure to build affordable housing.
3. **Bank Risk:** Banks should be careful with luxury properties, as our model showed higher error rates there (harder to value).

5 Conclusion

We compared 11 algorithms and found that simple Ridge Regression works just about as well as complex Boosting models for this dataset ($R^2 \approx 0.62$). This challenges the idea that "more complex is always better." For Almaty, a simple, fast linear model is actually a great choice for an automated valuation system, as long as you have good location data.

6 Future Work

- **Currency:** We should add the KZT/USD exchange rate since it affects secondary market prices.
- **Earthquake Safety:** After recent quakes, newer buildings might be worth more. We want to scrape "Seismic Resistance" data next time.
- **Air Quality:** Adding winter pollution data per district could help explain why some areas are cheaper.

References

1. Rosen, S. (1974). Hedonic prices and implicit markets: product differentiation in pure competition. *Journal of Political Economy*, 82(1), 34-55.
2. Park, B., & Bae, J. K. (2015). Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data. *Expert Systems with Applications*, 42(6), 2928-2934.
3. Tulemissov, A., Abdallah, W., & Marat, R. (2022). Spatial analysis of real estate market in Almaty: A machine learning approach. *Central Asian Economic Review*, 5(2), 112-125.
4. Nurgaliyev, D. (2023). Neural network applicability in volatile markets: Evidence from Astana. *Kazakhstan Journal of Applied Mathematics*, 12(4), 88-101.
5. Wang, X., Zhang, Y., & Chen, H. (2023). A comparative study of GBDT algorithms for real estate valuation. *International Journal of Geographical Information Science*, 37(8), 1-22.
6. Sultanov, A., & Alibekov, A. (2023). Infrastructure valuation in post-Soviet cities: The impact of district heating. *Central Asian Journal of Economics*, 19(1), 45-60.
7. Kim, J., & Lee, S. (2024). Hybrid CNN-LSTM models for analyzing temporal volatility in developing real estate markets. *IEEE Access*, 12, 10567-10578.

8. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.
9. Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5), 1189-1232.
10. Ho, T. K. (1995). Random decision forests. *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, 278-282.
11. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785-794.
12. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30, 3146-3154.
13. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 6638-6648.
14. Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267-288.
15. Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 301-320.
16. Yrsltn. (2025). Housing Price Prediction Dataset (Kazakhstan 2025). Kaggle. Available at: `https://www.kaggle.com/datasets/yrsltn/housing-price-prediction-dataset-kazakhstan-2025` (Accessed: January 6, 2025).
17. AnsiNitro. (2025). AML Assignment 1 Code Repository. GitHub. Available at: `https://github.com/ansinitro/aml/tree/main/ass1`

Appendix A Detailed Algorithmic Reference

For anyone wanting to reproduce this, here are the formal definitions of the models we used.

A.1 Support Vector Machines vs. Tree Ensembles

SVMs try to find a dividing line (hyperplane). Tree Ensembles (like Random Forest) chop the data into boxes (hyper-rectangles). For Almaty housing, where "District A" is just totally different from "District B", trees work better naturally.

A.2 Gradient Boosting Implementation Details

- **Loss Function**: We used Squared Error loss $L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$.
- **Regularization**:
  - **XGBoost**: Uses γ and λ to stop trees from getting too complex.
  - **LightGBM**: Uses `min_data_in_leaf`.
  - **CatBoost**: Uses `depth` and `l2_leaf_reg`.

A.3 High-Performance Boosting Libraries Comparison

1. **XGBoost**: Optimized for systems. It uses all CPU cores and is super fast because of cache-aware access.
2. **LightGBM**: Grows trees "leaf-wise". It picks the leaf with the biggest error to fix. This can be more accurate but might overfit more.
3. **CatBoost**: Uses "Symmetric Trees". It splits the same way at each level, making it balanced and stable. It handles categories by turning them into numbers using statistics.

Appendix B Code Implementation Structure

The project is set up so it's easy to read:

- `src/train.py`: The main training script. We used `StandardScaler` inside the Cross-Validation loop to avoid data leakage.
- `src/preprocess_real_data.py`: A script that parses the raw text (like getting numbers from "3-komnatnaya").
- `report/figures/`: High-res images.

Appendix C Full Feature List

1. **Price (KZT)**: What we are predicting.
2. **Area (m^2)**: Size of the apartment. This was the biggest predictor.
3. **Rooms**: 1-6 rooms.
4. **District**: One-Hot Encoded variables:
    - Almaly, Medeu, Bostandyk, Auezov, Jetysu, Turksib, Alatau, Nauryzbay

Appendix D Glossary of Technical Terms

- **Average Marginal Effect (AME):** How much the prediction changes when you increase one variable by one unit.
- **Bagging:** Combining predictions from multiple models to reduce variance (making it more stable).
- **Boosting:** Combining models to reduce bias (making it more accurate).
- **Cross-Validation:** Splitting data into k parts to test the model on data it hasn't seen.
- **Heteroscedasticity:** When the errors vary widely (e.g., small errors for cheap houses, huge errors for expensive ones).
- **Hyperparameter Tuning:** Finding the best settings for the algorithm.
- **Multicollinearity:** When features are highly correlated (like "District A" and "Not District A").
- **One-Hot Encoding:** Turning categories into binary 0/1 columns.
- **Overfitting:** When a model learns the training data too perfectly and fails on new data.
- **R-Squared (R^2):** How much of the variance in price is explained by our model.
- **Regularization (L_1/L_2):** Penalizing complex models to prevent overfitting.
- **Residual:** The difference between the real price and our predicted price.
- **RMSE:** The average error of the model.