Final Project Report: Predicting Apartment Sale Prices

Niv sade 208639252

Tom regev 206836686

1. Introduction

In this final project, we applied machine learning techniques to predict apartment sale prices using various features. The project involved data cleaning, feature engineering, model selection, and evaluation. Our goal was to minimize the Root Mean Square Logarithmic Error (RMSLE) and improve the predictive power of our model.

This file is organized in the same order as the code.

2. Exploratory Data Analysis

2.1 Dataset Overview

- The dataset contains information about apartments, including numerical and categorical variables.
- The dataset contains a lot of missing values that we filled with different methods.

2.2 Descriptive Statistics

- **Numerical Features:** We calculated mean, median, standard deviation, min, and max values.
- **Categorical Features:** We examined frequency distributions and identified imbalances and patterns.

2.3 Visual Analysis

- **Histograms and Boxplots:** We used to detect skewness and outliers in a few parameters which helped us delete some of the observations.
- **Scatterplots:** We examined relationships between a few different parameters with the target parameter.
- **Correlation Matrix:** We identified strong and weak correlations.

3. Data Cleaning

3.1 Handling Missing Values

For start we changed some of the values we suspected to be fake to NA so we can predict them with our models and maximize the model training and performance later on.

- **Numerical Features:** Imputed missing values using mean, median, grouped average and model predictions:
    - For the first missing data handeling we came up with the idea to identify observations from the same building and fill the matching parameters such as max_floor, build_year and material and finding out crucial insights regarding the ratio in the same buildings such as full_sq, kitch_sq and life_sq, this method helped us predict about 60 precent of the crucial missing data.
    - For the next step we used XGBOOST to compute the other NA's in the data set.
- **Categorical Features:** Filled missing values using mode or left them as missing for tree-based models:
    - We added a new parameter that rely on the sub_area parameter and divide all the area to cheap area, medium price area and expensive area and transformed it to dummy variables.
    - We filled the missing values in material using the matching building method at first and then most common material.
    - We filled the state criteria using the most common value in different areas.
- **Timestamp Processing:** Converted timestamps to extract useful time-based features like inflation and other economic criteria to help determine the prices change threw the years and then tune the model to consider the date the transaction occur.

3.2 Outlier Detection and Treatment

- **Histogram and scatterplot:** identified extreme values and fake observations:
    - When creating the histogram of the prices in the training set we noticed that there is a lot of observations with unrealistic prices, after a quick check online we learned that a lot of houses transactions report a lower price because of tax fraud so we knew that those observations don't conflict the real data and we need to drop them, we set a lower boundary at 35,000 ruble per square meter and 500,000 for the upper boundary.
- **Boxplots and IQR Method:** Identified extreme values and capped/removed unrealistic data points:
    - For start we used boxplot and projecting it on the full_sq ranges (10-30, 31-50, etc) helped us determined the reasonable price range for different observations.
- **Domain Knowledge Checks:** Ensured reasonable values (for example filtering out unrealistic full_sq values > 500 m² or life_sq > full_sq, full_sq < 10).

4. Feature Engineering

4.1 Created Features with interaction

- **Price per Square Meter:** price_per_sq = price_doc / full_sq
- **Rooms per Square Meter:** room_size = full_sq / num_room
- **Building Age:** bought_minus_built = year_sold - build_year
- **Categorical Transformation:** One-hot encoding for categorical variables.
- **Area Price Classification:** Segmented sub_area into cheap_area, middle_area, and expensive_area.

While creating the features above we checked their correlation to the target parameter and how much they maximize the model accuracy to predict the prices.

4.2 Principal Component Analysis (PCA)

- Applied PCA to reduce dimensionality in spatial, economic, and demographic features.
- Selected the top components contributing to variance.
- Using PCA by category and correlation table to choose the best features to keep and avoid overfitting.

5. Data Splitting

- **Train-Test Split:** 80%-20% split to train and validate the model.
- **Separate Analysis for Investment vs. Owner-Occupier Homes:** Data divided based on product_type.

6. Model Development

6.1 Models Used

- **Tree-Based Models:** Random Forest, XGBoost
- **Feature Scaling:** Applied for distance-based models

  We started with linear regression, then lasso and ridge and we found out that XGBOOST did the best prediction model because it is a tree-based model that effectively handles non-linear relationships between variables, categorical features, and missing values, whereas linear models rely on the assumption of linearity and cannot capture complex feature interactions without advanced feature engineering.

6.2 Hyperparameter Tuning

- **XGBoost Parameters Tuned:**
- n_estimators: 400
- learning_rate: 0.1
- max_depth: 6
- min_child_weight: 1
- subsample: 0.8
- colsample_bytree: 0.9
- colsample_bylevel: 1
- reg_alpha: 0

- reg_lambda: 10
- seed: 42
- objective: 'reg:squarederror'
- nthread: 8

7. Model Performance & RMSLE Improvement

Rmsle reduction action	**Stage**	**RMSLE**
	Initial Baseline (linear regression)	0.5352
1. XGBoost	XGBooast modeling transformation	0.4652
2. Data cleaning	Removing outliers and exceptional prices	0.3343
3. Feature engineering	Feature engineering and feature selection	0.3274
	Model tuning	0.3222
	Data separation (investment and owner) and model tuning	0.3167

- **Final RMSE:** 0.3167 (XGBoost, tuned parameters)
- **Owner-Occupied RMSE:** 0.1122
- **Investment Property RMSE:** 0.3900

8. Conclusion

8.1 Key Insights

- Feature engineering significantly improved model performance.
- Creative thinking to fill missing values.
- PCA reduced redundancy in correlated variables.
- XGBoost outperformed other models with proper tuning.