Real Estate Pricerange Predictor

Github: https://github.com/GnuThe2nd/Real-Estate-Pricerange-Predictor.git

Team: Mari Lee Lumberg, Karl Martin Puna, Daniel Henri Trump

1.Business understanding

1. Identifying Business Goals

Background

This project focuses on developing a predictive algorithm for rental price ranges in Estonian cities, specifically Tartu and Tallinn. The system aims to predict rental prices for any given coordinate within city borders by analyzing scraped rental data from KV.ee (Estonia's most popular real estate website) combined with geographical data from OpenStreetMap (OSM). The project was initiated as part of a Tartu University Computer Science class, with the goal of creating a practical tool for understanding rental market dynamics in Estonia.

Business Goals

The primary stakeholders who will benefit from this project are:

- **Prospective renters** seeking to understand fair market prices in specific neighborhoods
- **Property owners** looking to set competitive rental rates
- **Students and researchers** studying Estonian urban housing markets

The core goal is to create an accessible algorithm that accurately predicts rental price ranges based on location coordinates, providing transparency in the rental market and helping users make informed decisions.

Business Success Criteria

Success will be measured by: The algorithm's ability to predict rental price ranges within an acceptable margin of error for at least 70% of test cases. Coverage of sufficient geographical areas in both Tallinn and Tartu to be practically useful. The model's capacity to identify and explain key factors driving price variations across different neighborhoods

2. Assessing the Situation

Inventory of Resources

Data Sources: Rental listings scraped manually from KV.ee, containing information such as price, location coordinates, square footage, floor information, and building build year. As well as geographical and infrastructure data from OpenStreetMap for mapping service locations

Technical Resources: Python programming environment with data science libraries (pandas, scikit-learn, etc). Jupyter notebooks for exploratory analysis and model development. Custom helper functions and methods for data processing and analysis.

Requirements, Assumptions, and Constraints

Key Constraints:

Uneven distribution of rental data across cities and neighborhoods. Some areas have significantly more listings than others, which may affect model reliability in data-sparse regions.

Inconsistent data completeness across listings. Not all rentals provide the same information fields, leading to missing values.

Manual scraping limitations. Automated scraping was abandoned due to KV.ee's API restrictions, making data collection time-intensive

Assumptions: The scraped data represents current market conditions reasonably well. Location coordinates provided by KV.ee are accurate. The relationship between location features and prices remains relatively stable over the project timeline.

Risks and Contingencies

Data Quality Risks: False or misleading information in listings (e.g., incorrect square footage, misrepresented amenities). Insufficient data volume in certain areas, potentially leading to unreliable predictions for those neighborhoods. Outdated listings that no longer reflect current market prices.

Contingency Plans: Implement data validation and outlier detection to identify suspicious entries. Clearly communicate confidence levels in predictions based on data availability. Focus analysis on areas with sufficient data density while acknowledging limitations elsewhere

Terminology

Price per square meter (€/m²): Primary metric for comparing rental values across different property sizes

Service proximity: Distance to amenities such as shops, schools, public transport, and entertainment

Floor level vs. total floors: Distinction between the floor number of a specific rental and the total number of floors in the building

Rental neighborhood clustering: Grouping of nearby rentals for comparative analysis

Costs and Benefits

Costs: Time investment in manual data collection and cleaning. Computational resources for model training and testing

Benefits: Enhanced transparency in Estonian rental markets. A Practical learning experience in applied data science and machine learning. A functional tool that can be extended to other Estonian cities or updated with new data and insights into which location factors actually drive rental prices (preliminary findings suggest proximity to other rentals is more significant than service proximity).

3. Defining Data-Mining Goals

Data-Mining Goals

The primary data-mining objective is to extract high-quality, relevant features from the scraped KV.ee data and OSM geographical information that can accurately predict rental price ranges. Specifically:

Clean and standardize rental listing data to ensure consistency

Engineer meaningful features from location coordinates and property characteristics

Identify which variables have the strongest correlation with rental prices

Build spatial relationships between rentals to capture neighborhood effects

Data-Mining Success Criteria

The data-mining effort will be considered successful if:

- We can identify at least 2-4 significant predictors of rental prices with statistical confidence
- The cleaned dataset retains at least 60% of originally scraped records after quality filtering
- Feature engineering produces variables that improve model performance compared to using raw data alone
- We can draw actionable, evidence-based conclusions about what drives rental price variations in Tartu and Tallinn - conclusions that go beyond generic assumptions and reflect the actual Estonian rental market dynamics

2. Data understanding

Gathering data

The objective is to analyze real-estate listings from kv.ee, Estonia’s largest property portal. We wish to understand price patterns, offering types and property characteristics. To support this goal, the dataset must include fields that allow price comparison and characterization of listings.

Outline data requirements

Each city we are looking at has some key columns we need, these include:

- **Metadata:** id, url to listing
- **Location:** longitude, latitude
- **Property characteristics:** total area meters squared, number of rooms, floor number, state of the rental (does it need fixing or not), if its an apartment, building year
- **Price:** total price

Verify data availability

All data in the dataset was obtained through web scraping directly from publicly visible kv.ee listing pages (manual scraping tutorial in our Github README). We did test automated scraping, but after a few hours of hardfought battles against kv.ee database API, we concluded that while yes, most likely possible, the actions we were taking started to stray into the gray area of what many would call “hacking” and with the API’s constant resistance, it is no longer worth automating the scraping for a task we most likely won't be doing again. We did consider trying to automate a script to open the website, click all the right buttons and pull the data that way, but again, we most likely wont need to scrape the data more than once or twice, so what's the point?

TL,DR: No automated scraping, not worth the time investment.

Define selection criteria

Selection criteria was quite straightforward: if the property lacked either the price, the area or the number of rooms column, the property was not selected. We also didn't pull data for any lands without a rental home, that means just empty land spaces do not count. We later also decided to scrap any properties that had over 20% of data missing, to try and balance the playing field, so to speak.

Describing data

We looked at data for 2 different cities, Tartu (448 listings) and Tallinn (1345 listings). Each row corresponds to a unique property listing, and each column represents one attribute extracted from the kv.ee page. What exactly was pulled, can be read above, under data requirements.

Exploring data

Our biggest concern was immediately, that Tartu has 3 times less listings then Tallinn, which while it makes sense, might actually make it impossible to predict the price ranges correctly. That aside, for Tartu the prices range from 70 - 2000 euk per month, and for Tallinn 150 - 7500 euk per month. We notices that as a general rule, for Tartu, the prices seemed to get higher the closer one was to the university buildings. Still Tartu lacks data in quite a few places, thus we shall see if it is possible to conclude anything from that dataset.

Verifying data quality

Since the data is pulled directly from the website and has gone through kv.ee data cleaning plus validation checks and our own cleaning too, all should be fine. We are lucky on that front :)

3. Planning your project

1. Scrape data from KV.ee. The data should only include rental listings that are within Tallinn and Tartu. Scraper was not built during this course.(Daniel Henri, ~3h)

2. Clean and get an understanding of the scraped data. Some columns had a lot of missing data. (Mari Lee, ~8h)
3. Format clean data into new files with separate versions for Tartu and Tallinn (Mari Lee, ~2h)
4. Get OSM data and clean it and aggregate it into categories. Aggregation is important because it makes the data easier to understand and reduces multicollinearity. (Daniel Henri, ~8h)
5. Cleaned data analysis. It is important for finding features that might be important when predicting price. Also helped in understanding the data and feature engineering. (Daniel Henri, ~16h)
6. Model building on both Tartu and Tallinn. We tried Random Forest with regular folds, Random forest with spatial folds. Tested different types of engineered features. Target encoding a grid of different sizes within spatial folds. Target encoded city districts within spatial folds. Binned the price ranges into 3 categories (budget, mid range, premium). Tested different hyperparameters. Settled on XGBoost model. (Daniel Henri, ~35h)
7. Visualizations. Visualised the results on maps where possible. Spatial distribution of some of the engineered features. Spatial distribution of the actual rent prices and the distribution of predicted rent prices. Map of the spatial accuracy of the model. (Daniel Henri, ~10h)
8. Separate data cleaning code and data analysis and model building code into two separate scripts. Important so the workflow is smoother, faster and more understandable. (Karl Martin, ~10h)
9. Team meetings and discussions (All ~6h, 3 meetings, 2h each)
10. Try to make data scraping automatic. Did not work out (Karl Martin, ~2h)
11. Make a master file. Important so our work is documented. (Karl Martin, ~10h)
12. Make the poster. (Mari Lee ~10h, Others ~1-2h)