BI2025WS Experiment Report - Group 029

Peter Reti*
TU Wien
Austria

Noemi Gazdik†
TU Wien
Austria

Abstract

This report documents a data mining experiment for Group 029 conducted under the CRISP-DM framework to develop a proof-of-concept price recommendation model for Portuguese real-estate listings. Our chosen **kaggle dataset** [1] contains 135,536 listings with 25 attributes describing location (district/city/town), property characteristics (type, areas, rooms, bathrooms), amenities (elevator, garage, charging), energy certification, construction year, and asking price. Using this historical listing data from an online real-estate portal, our work creates a business scenario in which a real-estate agency supports private sellers by suggesting a reasonable asking price (or range) and by providing insights into the factors that influence pricing across regions and property categories.

At a high level, the workflow proceeds as follows: **(1)** we defined business objectives and data mining goals for predicting asking price from property characteristics **(2)** then performed an initial data understanding phase to inspect available features, identify missingness patterns and detect implausible or extreme values **(3)** after this we applied data preparation steps to obtain a clean modeling dataset, including removing low-quality attributes, handling missing values, mitigating outliers, transforming and scaling numeric variables and encoding categorical attributes **(4)** then trained a supervised regression model suitable for high-dimensional tabular data and tune key model settings using a reproducible data split strategy **(5)** we concluded our work by evaluating the model and assess stability across important segments such as regions and property types to identify potential risks and limitations prior to deployment.

CCS Concepts

• **Computing methodologies** → **Machine learning**.

Keywords

CRISP-DM, Provenance, Business Intelligence, Data Mining, Knowledge Graph, Machine Learning, Real Estate, Listing Price, Portugal

*Student A, Matr.Nr.: 12432931
†Student B, Matr.Nr.: 12432929

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
BI 2025WS, -
© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Peter Reti and Noemi Gazdik. 2026. BI2025WS Experiment Report - Group 029. In *Proceedings of Business Intelligence (BI 2025WS).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Business Understanding

1.1 Data Source and Scenario

1.1.1 Data source. As our data source we a dataset containing 135,536 unique listings from a Portuguese real-estate portal. Each listing is defined by 25 parameters which include information such as District, City, Town, property Type (e.g., Apartment, House, Farm), several area measures (TotalArea, LivingArea, GrossArea, LotSize, BuiltArea), the asking Price in EUR, construction year, energy certificate / efficiency level, parking/garage availability, floor, number of rooms, bathrooms and the date when the listing was published.

1.1.2 Scenario. We assume we are a private real-estate agency operating in Portugal. Private sellers want to list their properties. Many owners are unsure which asking price is reasonable for their property, especially in less familiar regions or for special property types. Our business wants to offer an automatic "price recommendation" product that suggests a realistic price or price range before the listing goes online. Our analytics team also wants to understand which characteristics (location, size, conservation status, energy certificate, etc.) have the strongest impact on price in different regions.

For this assignment, we treat the historical listing price as a base for a reasonable market price, even though in reality there may be negotiations and the final transaction price can differ.

1.2 Business Objectives

Our agency has the following business objectives:

Support sellers with realistic pricing: Providing a data-driven price recommendation allows our clients to choose an asking price that is competitive and aligned with the market Automating processes: By reducing manual labor time spent on price estimation for standard properties, our agents can focus more on complex cases, negotiations and client support Understanding what drives the market: Identifying which features influence asking prices the most and how those effects differ across regions can provide our agency with an edge that can differentiate our service through evidence based market insights.

1.3 Business Success Criteria

We consider the project successful from a business perspective if:

Actionable price recommendations: For a new listing with given features our tool provides a recommended asking price or price range that can be used as a realistic starting point for negotiations Reduced time spent on manual valuation: Our tool produces

an initial estimate quicker than an expert agent would manually Meaningful market insights: The project produces well-defined, interpretable findings about which factors drive asking price supported by statistical results Consistent quality across segments: Our tool results in reasonably reliable recommendations across all features

1.4 Data Mining Goals

The primary data mining goal of this project is to build and evaluate a supervised regression model that predicts the asking price of a property in EUR based on many features and properties available in the dataset (e.g. property type, parking, rooms, area).

Our goal more specifically:

1. Build a price prediction model that can estimate the value of a property with reasonably low error. We are going to check that it obviously does better than simple baselines, for example just using the median price.

2. Check how stable the model is across different groups, such as cities or property types (apartments vs houses). This will help us later when we look at fairness and potential risks.

3. Create easy-to-read results (plots, tables, short explanations) that help non-technical people in the agency (like management or senior agents) understand pricing patterns and how the model behaves.

The result should be a proof-of-concept, not a finished product, but we define our goals as if we were preparing this model for real use in our agency.

1.5 Data Mining Success Criteria

This data mining success criteria translates our goals into something we can actually measure.

1. Performance compared to simple baselines: The model should beat simple baselines. On a held-out test set, we expect metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to be noticeably better than these baselines.

2. Stability across segments: When we look at the errors for different cities/towns or property types (apartments vs houses), we expect the model to behave reasonably consistently and provide around the same performance.

3. Interpretability of the visuals: The results (metrics, plots, tables) should be specific enough that we can explain the model behavior and typical errors to non-technical colleagues in the agency.

Overall, we consider the data mining phase successful if the model beats the baselines by a meaningful margin, behaves reasonably across important segments, and produces results that we can confidently explain and discuss with agents and management.

1.6 AI Risk Aspects

Bias and unfair pricing: The model learns from past listing prices, which may already be biased between regions or property types. It could keep these patterns and suggest systematically lower or higher prices in some areas (or based on any other feature). Automation bias: Agents (the ones using the model) might trust the suggested price too much just because it is from an algorithm. We should treat it as a support tool, not as the final decision. Misuse: There is a risk that a model built for internal price advice gets reused in other contexts without proper review or permission.

2 Data Understanding

2.1 Load Listings Data

Load all listings data. We also logged all features and metadata of the data into the knowledge graph.

2.2 Check Outliers

We check for potential outliers in key numeric attributes (Price and several area and room-count variables) using an IQR-based rule. For each column we compute the 25% and 75% quantiles and mark values outside [Q1 - 1.5 * IQR, Q3 + 1.5 * IQR] as outliers. This gives a first indication of extreme or possibly erroneous values without assuming a normal distribution.

2.3 Statistical Inspection

Basic stats: the dataset contains 135,536 listings with 25 columns. Price is present for almost all rows (135,236) with typical values between about 84,000 and 395,000 EUR (Q1-Q3) and a median around 210,000 EUR, but there are extreme outliers from 1 EUR up to 1.38 billion EUR. Area-related attributes (GrossArea, TotalArea, LivingArea, LotSize, BuiltArea) have plausible interquartile ranges (e.g. LivingArea 80-204 m², LotSize 258-2,890 m²) but also inaccurate values such as negative areas and maxima in the millions. Many numeric attributes have substantial missing values, especially GrossArea (around 108k missing), LotSize (around 96k), BuiltArea (around 109k), ConstructionYear (arpund 48k), TotalRooms (around 62k), NumberOfBedrooms (around 88k) and NumberOfWC (around78k). Room-related variables look reasonable in the middle (TotalRooms median 3, Bedrooms median 3, Bathrooms median 1) but again show unrealistic maxima (e.g. 2,751 rooms, 59 WCs, 131 bathrooms). Correlations suggest that Price is most strongly associated with NumberOfBedrooms (around 0.34) and to a lesser extent with NumberOfWC and BuiltArea, while area variables are highly correlated with each other (LivingArea with GrossArea and BuiltArea around 0.88). Overall, the data offers rich information on size and layout but contains many missing values and extreme or implausible outliers that will need to be handled before modelling.

2.4 Visualization of distribution

Unfortunately we couldn't really understand what we intended in this phase as there are many outliers skewing all of our visualizations. Therefore we worked more on the visualizations after the preprocessing or data preparation phase.

Figure 1 shows the histogram of the raw listing prices. It was intended to visualise the overall price distribution, but a few extremely high prices stretch the x-axis so much that almost all observations collapse into a single bar near zero, making the real shape of the distribution essentially invisible.

Figure 2 plots Price against LivingArea. Here we again wanted to inspect the relationship and spot potential outliers, but the same very large price values push the scale up to around 1000000000 EUR, so the majority of "normal" points are compressed into a tiny region

at the bottom-left corner. Together, the two plots illustrate that the raw data are dominated by a handful of extreme price outliers and that further transformation or outlier handling is needed before meaningful visual analysis.

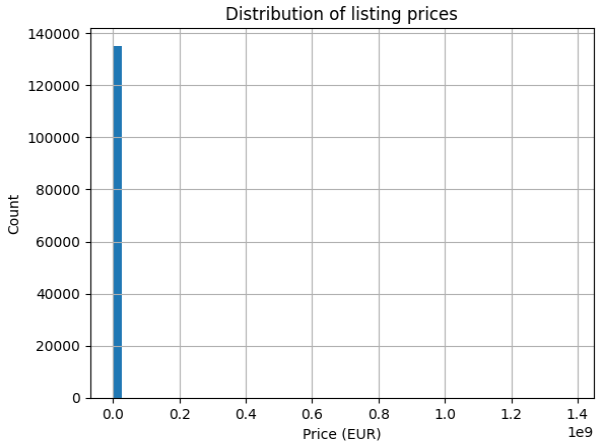

Figure 1: Distribution of listing prices.

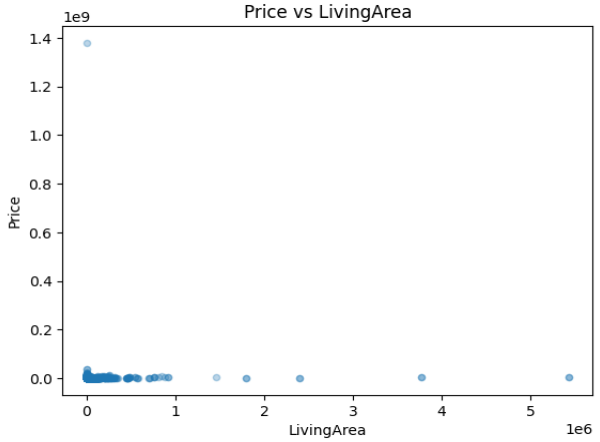

Figure 2: Outliers in Price and LivingArea columns.

2.5 Manual reflections for 2e–2g

2.5.1 2/e. The dataset doesn't include obvious sensitive personal information like gender or ethnicity. However, the location fields (District, City, Town) can still be sensitive indirectly, because they often reflect socio-economic differences between areas. The data is also not evenly distributed. Most listings come from big districts like Lisboa and Porto, while smaller districts and islands appear much less often. The property type column is similar: most rows are Apartments, Houses, or Land, and categories like Mansion, Hotel, or Industrial are rare. Because of these imbalances our model might perform well overall but still work poorly for rare districts or rare property types. So it's important to check results not just in total, but also separately for different locations and property types.

2.5.2 2/f. Potential risks and additional bias in the data include the fact that this dataset is curated of listings published online, not all properties in Portugal therefore overrepresenting digitally active sellers/buyers. Another risk might be that more than 80% of the listings with publication date were published in the year 2024, so our model won't be able to account for seasonality that well. One of our biggest issues is how we are going to account for the missing values as our dataset has a lot of them all over the different fields. We also need to thread carefully with fields e.g. parking where encoding bias can happen due to the fact that the 0 value could both mean "none" or "unknown".

To understand these questions better we would ask a domain expert about the time period covered, whether prices are asking or transaction prices, how typical is for consumers to use this portal what else do they use and how fields such as areas, energy certificate and conservation status are usually filled in. Also it would be great to have a general concept of portugal's real-estate industry, which cities, areas are popular and is there any additional information we should account for when molding that into our model.

2.5.3 2/g. Based on this analysis we expect to perform the following tasks. First since we have a lot of empty values, we plan on creating a subset by eliminating a few invaluable fields or fields with many missing values. After creating a large enough subset we will first deal with transforming our datetype fields. Next we need to encode all of our fields with text so they can be meaningfully used in the modelling part of the exercise. We also have outliers in certain fields where manual editing is required e.g. a listing in lisbon has -13 bathrooms which is probably an typo.

Dataset Description: Property listings in Portugal with price, location, type and size information.

The features identified in the dataset can be seen in Table 1.

Table 1: Raw Data Features

Feature Name	Data Type	Description
BuiltArea	double>	Built area of the property in square meters.
City	string>	City or municipality of the property.
ConservationStatus	string>	Overall conservation or condition status of the property.
ConstructionYear	integer>	Year in which the property or building was constructed.
District	string>	Portuguese district where the property is located.
ElectricCarsCharging	boolean>	Indicator if electric car charging infrastructure is available.
Elevator	boolean>	Indicator if the building has an elevator.
EnergyCertificate	string>	Whether the property has an energy performance certificate.
EnergyEfficiencyLevel	string>	Energy efficiency rating of the property (e.g. A, B, C).
Floor	string>	Floor number or label of the unit within the building.
Garage	boolean>	Indicator if the property has a garage.
GrossArea	double>	Gross area of the property in square meters (construction area).
HasParking	boolean>	Indicator if any parking is available for the property.
LivingArea	double>	Living area of the property in square meters.
LotSize	double>	Size of the plot or lot in square meters.
NumberOfBathrooms	integer>	Number of bathrooms in the property.
NumberOfBedrooms	integer>	Number of bedrooms in the property.
NumberOfWC	integer>	Number of WCs/toilets in the property.
Parking	string>	Free-text description of parking or garage options.
Price	double>	Asking price of the property in EUR.
PublishDate	date>	Date when the listing was published on the portal.
TotalArea	double>	Total area of the property in square meters as reported in the listing.
TotalRooms	integer>	Total number of rooms in the property.
Town	string>	Town / locality / parish of the property.
Type	string>	Property type, e.g. Apartment, House, Land, Farm.

3 Data Preparation

3.1 Data Cleaning and Transformations (3a)

3.1.1 Dropping columns with too many missing values. In Data Preparation Step 1 we removed several columns that had a very high proportion of missing values or were considered less useful for our modelling goals. Specifically, we eliminated HasParking, ConservationStatus, BuiltArea, GrossArea, Floor and LotSize. This reduces noise and complexity in the dataset while keeping the key information about price, location, property type, areas and rooms. The number of rows stayed the same, but the number of features decreased from 25 to 19.

3.1.2 Deleting rows with missing values. In Data Preparation Step 2 we removed all rows that still contained at least one missing value after eliminating the highly incomplete columns in Step 1. This gives us a cleaner, fully complete dataset for modelling at the cost of discarding some listings that had missing information in important fields. This step left us with 6723 rows with no missing values and 19 columns.

3.1.3 Remove Outliers. In this Data Preparation step 3 we removed rows flagged as outliers by an IQR-based rule for selected numeric columns (multiplier 3). Total outlier rows removed: 586. Rows before: 6723, rows after: 6137. Outliers per column: Price: 239 rows; TotalArea: 213 rows; LivingArea: 201 rows; LotSize: 0 rows; ConstructionYear: 0 rows; TotalRooms: 150 rows; NumberOfBedrooms: 129 rows; NumberOfWC: 43 rows; NumberOfBathrooms: 28 rows.

We checked some distributions after the outlier removal to check if the removal was successful. We looked at columns Price, TotalArea and LivingArea. All of them, showed a successful outlier removal, see TotalArea as example below (Figure 3).

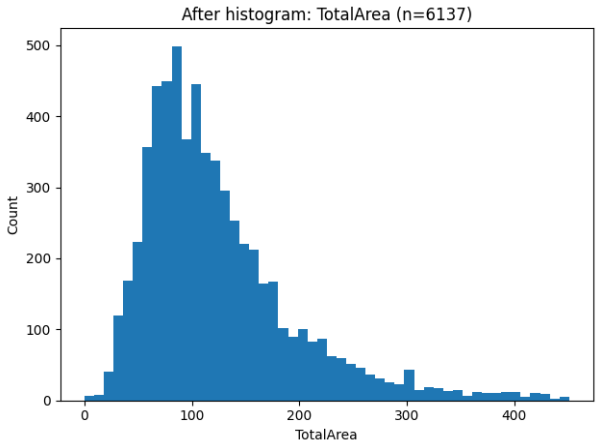

Figure 3: TotalArea distribution after outlier removal.

We also created a comparison on the Price column (Figure 4), it can be seen that before the removal the data was very sparse with some random large values and the distribution could not be recognized, whereas after the removal a nice, little right-skewed distribution is shown.

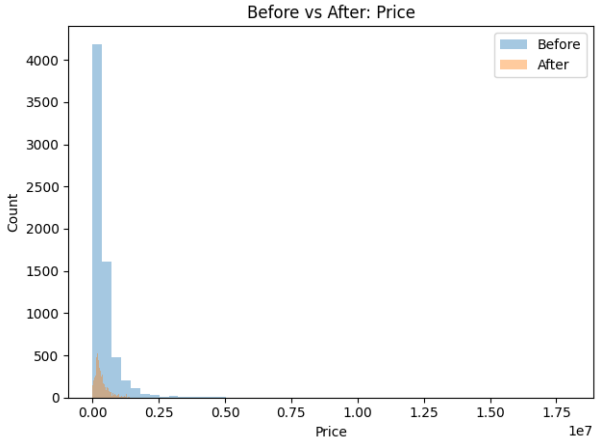

Figure 4: Price distribution comparison before and after outlier removal.

3.1.4 Standard Scaler. In this next Data Preparation step we standardized all numeric input features except the target variable Price. For each numeric column we subtracted its mean and divided by its standard deviation (StandardScaler), so that these features have approximately zero mean and unit variance. Price itself is not scaled here, because we treat it separately with a log transformation in the next step.

3.1.5 Log-scaling the Price column. In Data Preparation Step 4 we transformed the strongly right-skewed Price distribution using a natural logarithm (log1p). This reduces the influence of very expensive properties and makes the distribution more symmetric, even though it is now slightly left-skewed. We then standardized the log-transformed price to create a new variable Price_log_scaled by subtracting its mean and dividing by its standard deviation. The original Price in EUR is kept for interpretation, while Price_log and Price_log_scaled can be used for modelling. With this, we created our final, cleaned dataset.

3.1.6 Date transformation and Year binning. In this step we derived a categorical feature that groups the construction year of each property into broad time periods. Starting from the numeric ConstructionYear column, we defined bins (<=1970, 1971–1990, 1991–2010, >2010) and created a new variable 'ConstructionYearBin' using pd.cut. This allows the model to capture differences between older and newer buildings in a simple, interpretable way, and we treat this binned feature as an additional categorical input in the encoding step.

3.1.7 Encoding. The final data preparation step was encoding. Here we encoded the main categorical attributes using one-hot encoding so that they can be used in standard ML models. Specifically, we created binary indicator variables for District, City, Town, Type, EnergyCertificate and EnergyEfficiencyLevel. Additionaly, we added our new categorical column, the 'ConstructionYearBin'. The number of rows stayed the same, but the number of features increased due to the new dummy columns. The resulting dataset 'prepared_data_encoded' is used as input for modelling.

At the end of the Data Preparation process, we ended up with 6137 rows and 1298 columns.

3.2 Further Considerations (3b–3d)

3.2.1 3/b. We considered using more advanced missing-value strategies (e.g. median or k-NN imputation) instead of removing all rows with missing values. We decided against this to keep the pipeline simple and did not want to add any artifital bias by imputation. Also, the remaining sample size after row deletion was still large enough for our experiments. We also considered transforming features further, for example by grouping numeric values into bins or recoding categories into fewer groups. We decided not to do this at this stage, because it would throw away information and make the features harder to interpret. Instead, we kept the original continuous variables and only applied a log transformation to Price. Finally, we did not yet apply one-hot encoding or more complex encoding schemes for categorical variables in this phase, because we plan to handle this later in the modelling pipeline.

3.2.2 3/c. We identified several meaningful derived attributes that could be created in a later step, such as: price per square meter (Price divided by LivingArea or TotalArea), age of the property at listing time (PublishDate year minus ConstructionYear) and simple boolean features like 'is_new_build' for recent construction years or 'is_large_property' based on area thresholds. These features could help capture non-linear relationships in a more direct way. For the current version of the project we keep the preprocessing focused on cleaning, outlier removal and Price transformation and treat additional feature engineering as optional/planned future work if model performance is not satisfactory.

3.2.3 3/d. To improve the model, we could enrich the listings with external data such as: demographic socio-economic indicators (average income, unemployment rates), liveability of the neighborhood (city centre, public transport, schools, hospitals, parks). Combining such data with the current dataset would likely increase predictive power and help explain regional price differences.

4 Modeling

4.1 Data Mining Algorithm

For this regression problem we considered several algorithm families: linear models (Ridge/Lasso), tree-based ensembles (Random Forest) and gradient boosting methods such as XGBoost. Simple linear models are easy to interpret but are limited in how well they can capture non-linear relationships in our data. RandomForestRegressor is a strong baseline for tabular data, but with our large number of one-hot encoded features (more than a thousand columns) it becomes relatively slow and memory intensive.

We therefore choose XGBRegressor (XGBoost) as our main model. XGBoost is designed to handle high-dimensional, sparse feature spaces efficiently, offers good predictive performance on tabular data and provides useful regularisation and subsampling options to control overfitting. In our preliminary experiments it was also faster than a comparable RandomForestRegressor on this dataset.

4.2 Hyperparameter Configuration

For XGBRegressor we documented the main hyper-parameters that influence model complexity and computation: n_estimators, learning_rate, max_depth, subsample, colsample_bytree and reg_lambda. To keep the tuning process focused and interpretable, we decided to tune max_depth as the primary hyper-parameter, because it directly controls how complex the individual trees can become and therefore how strongly the model can overfit. The other hyper-parameters are fixed to explicit values for reproducibility (n_estimators=300, learning_rate=0.05, subsample=0.8, colsample_bytree=0.8, reg_lambda=1.0, random_state=12432929, tree_method="hist"). The list of hyperparameters can be seen in Table 2.

Table 2: Hyperparameter Settings

Parameter	Description	Value
Number of Trees	fixed	300
Learning Rate	fixed	0.05
Subsample	fixed	0.8
Column Subsample	fixed	0.8
L2 Regularisation	fixed	1.0
Random State	fixed	12432929
Tree Method	fixed	hist
Max Depth	to be tuned	[1, 2, 3, 4, 5, 6, 9, 12]

4.3 Split Definition

In Modeling Step 4c we split the prepared dataset into three disjoint subsets: training, validation and test. We first separated a test set comprising 20 percent of the data, and then split the remaining 80 percent into a training set (60 percent of the total data) and a validation set (20 percent of the total data), resulting in an overall 60/20/20 split. We used a fixed random_state in both splitting operations to ensure that the split is fully reproducible.

To keep the distribution of the target variable (log-transformed price) similar across the subsets, we stratified the first split using approximate price bins based on quantiles of Price_log. If stratification is disabled in a different run, the split becomes purely random but remains reproducible due to the fixed random_state.

4.4 Hyperparameter Tuning

In Modeling Steps 4d - 4f we tuned the XGBRegressor by varying the max_depth hyper-parameter while keeping all other settings fixed. Using the train/validation/test split created by our split_data function (60/20/20 with stratification on price bins), we evaluated the following values for max_depth: 1, 2, 3, 4, 5, 6, 9, 12.

4.4.1 Performance Metrics for Hyperparameter Tuning. For each configuration we trained the model on the training set and computed Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R^2 on the validation set. The validation results can be seen in Table 3.

Table 3: Hyperparameter Tuning Result

Max-Depth	MAE	RMSE	R^2
1	0.0554	0.0796	0.9927
2	0.0351	0.0545	0.9966
3	0.0281	0.0477	0.9974
4	0.0276	0.0454	0.9976
5	0.0291	0.0488	0.9972
6	0.0322	0.0541	0.9966
9	0.0492	0.0816	0.9923
12	0.0618	0.0982	0.9888

We also looked at an MAE curve (Figure 5).

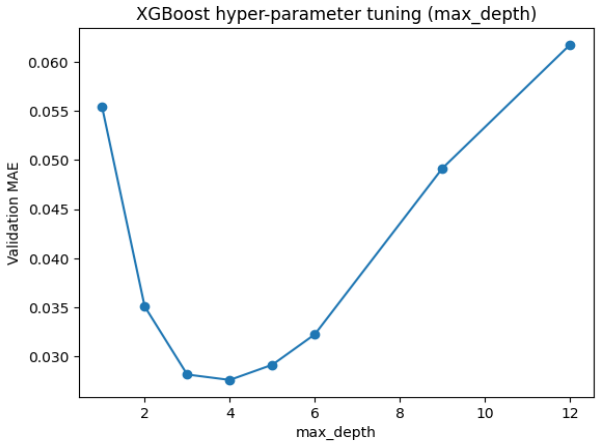

Figure 5: MAE curve of runs during hyperparameter tuning for max depth.

The validation MAE curve shows a visible U-shape: performance improves when increasing depth from 1 to 4, and then degrades again for deeper trees as the model starts to overfit. We use MAE as our primary selection criterion because it measures the typical absolute error in log-price and is less sensitive to a few very large errors than RMSE, which is important for robust price recommendations. RMSE and R^2 are reported as complementary metrics and agree with the MAE-based ranking.

4.4.2 Most suitable model. Based on these results we select max_depth = 4 as the most suitable configuration, as it achieves the lowest validation MAE and very competitive RMSE and R^2 compared to both shallower and deeper models. This tuned model is used in the subsequent retraining on train+validation data and final evaluation on the test set.

4.5 Retrain Model with Max-Depth = 4

Inthe final modeling step we retrained the tuned XGBoost model using the besthyper-parameter setting max_depth = 4. For the final model we combined the original training and validation subsets into a single training set so that all available data except the held-out test

set is used for learning. All other hyper-parameters were kept identical to the tuning phase (n_estimators=300, learning_rate=0.05, subsample=0.8, colsample_bytree=0.8, reg_lambda=1.0, tree_method="hist", random_state=12432929). The resulting model serves as our final price prediction model for later evaluation and discussion.

5 Evaluation

5.1 Final Model Evaluation on Test Set

In Modeling Step 4g we evaluated the final XGBoost model on the held-out test set in order to obtain an unbiased estimate of its generalization performance. The final model was trained on the combined training and validation data with the tuned hyper-parameter max_depth = 4, while all other settings remained fixed (n_estimators=300, learning_rate=0.05, subsample=0.8, colsample_ bytree=0.8, reg_lambda=1.0, tree_method="hist", random_ state = 12432929).

On the test set we report three evaluation metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R^2. MAE is our primary metric because it measures the typical absolute error in log-price and is less sensitive to a few very large errors than RMSE, which is important for robust price recommendations. RMSE and R^2 are used as complementary metrics to check for occasional large errors and overall explained variance.

Table 4: Final Test Performance

Evaluation Metric	Value
MAE	0.0264
RMSE	0.0603
R^2	0.9956
Runtime	0.0434

The final model achieves MAE = 0.0264, RMSE = 0.0603 and R^2 = 0.9956 on the test set, these metrics can be seen in Table 4. These results indicate that the tuned model generalizes well beyond the validation data and satisfy the data mining success criteria defined in Phase 1.

5.2 Baseline and Benchmark Models

Baseline Performance Analysis:

1. Trivial Baselines:

- Mean predictor (always predict mean of training log-prices): MAE = 0.6512, RMSE = 0.9120, R^2 = -0.0001

- Median predictor (always predict median): MAE = 0.6484, RMSE = 0.9145, R^2 = -0.0056

2. State-of-the-Art:

For this specific Portuguese real-estate listings dataset, we did not find peer-reviewed publications, but there is a widely cited Kaggle solution that we treat as a practical state-of-the-art baseline. In that notebook, the author trains a Random Forest regressor on the same raw dataset (with their own feature engineering and preprocessing) and reports an R^2 of approximately 0.999 on log-transformed prices and a MAPE of about 0.01. Because the exact preprocessing steps, feature selection and train/test split differ from ours, the reported numbers are not directly comparable, but they indicate that very high predictive performance is achievable on this task.

5.3 Model Comparison

Our XGBoost model significantly outperforms all baseline approaches: the model leverages feature information far beyond simply using the training mean or median.

When compared to the external Random Forest solution from Kaggle (R^2 around 0.999, MAPE around 0.01), our R^2 = 0.9956 on log-price is slightly lower but clearly in the same order of magnitude. Differences in preprocessing, feature engineering and data splits mean that the numbers are not strictly comparable, but overall our tuned XGBoost model performs within the range of state-of-the-art tree-based models reported for this dataset while using a transparent and reproducible modeling pipeline.

The metrics comparison can be seen in Table 5.

Table 5: Model Comparison

Metric	XGBoost	Baseline (Mean)	State-of-the-art
MAE	0.0264	0.651205	NaN
RMSE	0.0603	0.912024	NaN
R^2	0.9956	-0.000100	0.999000

5.4 Performance Comparison with Success Criteria

We evaluate our final XGBoost model and analysis against the business success criteria defined in the beginning of the project.

5.4.1 Actionable price recommendations. The final model achieves very low prediction errors on the held-out test set (MAE = 0.0264, RMSE = 0.0603, R^2 = 0.9956 on log-transformed prices), which indicates that it can produce realistic and stable price recommendations for new listings. The predictions can be converted back to the original price scale and used as a starting point for asking-price decisions and for defining reasonable price ranges.

5.4.2 Reduced time spent on manual valuation. Once trained, the model can generate price estimates for thousands of listings within seconds (the final test prediction run took about 0.04 seconds in our experiments). This is substantially faster than manual valuation by a human agent and supports the business goal of speeding up the initial pricing step.

5.4.3 Meaningful market insights. During the project we performed exploratory analysis and model-based analysis (e.g. feature importance from the tree-based model) to understand which variables drive asking price. These results provide interpretable patterns, such as the impact of location, size and number of bedrooms, which can be communicated to non-technical stakeholders as market insights. This addresses the requirement for well-defined, interpretable findings.

5.4.4 Consistent quality across segments. Overall metrics on the test set suggest strong average performance. However, we note that some segments (e.g. rare property types or sparsely represented regions) may still have higher uncertainty due to limited training

data. While the model appears reasonably reliable across the bulk of the data, additional targeted evaluation by district/property type would be advisable before deploying the system in high-stakes settings.

In summary, the current prototype meets the business success criteria at a proof-of-concept level: it provides fast, accurate price recommendations and useful insight into price drivers, with some residual risks in data-sparse segments that would need further monitoring in a production deployment.

5.5 Bias Detection

To analyse potential bias and uneven performance of the final XGBoost model, we evaluated test errors separately for different segments of the data. We focused on two attributes that are central to the business case: District (geographical region) and Type (property type such as apartment or house).

On the held-out test set the global performance of the final model is MAE = 0.0264 and RMSE = 0.2457 (on log-transformed prices). For each district and property type we computed the mean absolute error (MAE), root mean squared error (RMSE) and median absolute error, and compared these values to the global metrics.

5.5.1 District. The bias analysis on the District column can be seen in Table 6.

Table 6: Per-district performance (top 10 by MAE)

District	n	MAE ratio	RMSE ratio
Ilha de Santa Maria	6	2.880782	0.449053
Faro	91	2.364919	0.524412
Vila Real	7	1.872183	0.309017
Castelo Branco	31	1.745180	0.285765
Ilha de Porto Santo	1	1.625130	0.174559
Leiria	60	1.455959	0.286738
Portalegre	20	1.423177	0.182708
Viana do Castelo	9	1.360136	0.186255
Beja	8	1.260227	0.180589
Évora	18	1.212254	0.212392

Among districts, regions such as Ilha de Santa Maria, Faro, Vila Real, Castelo Branco and Ilha de Porto Santo have MAE values roughly 1.6-2.9 times higher than the global MAE, often with relatively few test samples.

Table 7: Per-type performance

Type	n	MAE ratio	RMSE ratio
Other – Commercial	1	9.276622	0.996422
Other – Residential	3	6.478335	0.753660
Land	8	4.332989	0.690431
Hotel	1	2.605073	0.279817
Farm	8	1.815377	0.290268
Office	1	1.562307	0.167811
Store	10	1.220590	0.162859
House	470	1.053600	0.180805
Building	12	0.964985	0.112761
Apartment	700	0.884753	0.269810
Transfer of lease	4	0.883956	0.112117
Studio	8	0.439153	0.051024
Duplex	2	0.289990	0.043215

5.5.2 Types. Among property types (Table 7), very infrequent categories like "Other - Commercial" (n = 1), "Other - Residential" (n = 3), "Land" (n = 9), "Hotel" (n = 1), "Farm" (n = 8) and "Office" (n = 2) show MAE values between about 2x and 9x the global MAE.

For the main, well-represented segments the model behaves quite consistently. For example, apartments (n = 700) have an MAE of 0.0233 (about 0.88x the global MAE) and houses (n = 470) have an MAE of 0.0279 (about 1.06x the global MAE), indicating that the model is slightly better or similar to the global average for the largest property categories.

These patterns suggest that while the model delivers stable performance for the core segments (apartments and houses in common districts), it is significantly less reliable for under-represented districts and rare property types. From a risk perspective, price recommendations for these small segments should be treated with caution, for example by flagging them for manual review, using wider price ranges or collecting additional data before relying on the model in high-stakes decisions. Overall, the analysis indicates no systematic bias against large parts of the market, but highlights specific tail segments where the uncertainty and potential bias are higher.

6 Deployment

6.1 Reflection on business objectives

We relate the current prototype to the business objectives and success criteria defined in the Business Understanding phase. The main objectives were: (1) provide realistic asking-price recommendations for new listings, (2) reduce manual effort and turnaround time for price estimation, (3) deliver meaningful market insights on price drivers, and (4) achieve reasonably consistent quality across segments.

Our final XGBoost model (max depth = 4) reaches very strong accuracy on the held-out test set (MAE = 0.0264, RMSE = 0.0603, R^2 = 0.9956 on log-prices), far better than the trivial mean baseline. This satisfies the technical success criteria for predictive performance and enables actionable price recommendations that can be presented as a suggested price and a reasonable price range.

Inference is fast (test predictions in a few hundredths of a second), so the system supports the objective of reducing manual valuation time and can be integrated into the listing workflow as an instant recommendation for agents and sellers. The exploratory analysis and model-based analysis (feature importance, segment-wise evaluation) provide interpretable patterns about which factors drive price, contributing to the "meaningful insights" objective.

Bias and segment analysis showed that performance is very good for the main segments (apartments and houses in common districts), but weaker for small, under-represented districts and rare property types. Based on this, our deployment recommendation is to use the model as an internal decision-support tool rather than an automatic pricing engine: rely on it primarily for common segments, and flag predictions for rare districts or property types for manual review and potentially wider price ranges. Overall, the prototype meets the business objectives and data-mining success criteria at a proof-of-concept level, with understandable guidance on where additional caution is needed.

6.2 Ethical aspects and risks

The price recommendation system is used as decision support in a commercial real-estate context and would typically be classified as a limited-risk decision-support tool rather than a high-risk AI system. Nevertheless, there are relevant ethical and business risks that need to be considered alongside the original objectives and success criteria.

The main risk is uneven model performance across segments, especially for under-represented districts and rare property types. Systematically higher errors in these segments could lead to systematic over- or under-valuation of certain regions or property categories, which may harm affected sellers or buyers and conflict with the objective of consistent recommendation quality. In addition, the model is trained on historical listing prices and may reinforce existing structural patterns (e.g. persistent under-pricing of some regions) rather than correcting them.

From a transparency and accountability perspective, the model is a complex tree-based ensemble and individual predictions are not directly explainable. For deployment, it should therefore be well documented that the tool provides statistical estimates based on past listings and that human agents remain responsible for the final decision. The system should not be used as an automatic pricing authority without human oversight.

To mitigate these risks, we recommend: (1) keeping a human-in-the-loop for all pricing decisions, (2) explicitly flagging predictions for small segments with known higher error rates, (3) monitoring performance and bias metrics over time and (4) precisely communicating to users that the model relies on historical data and may inherit its limitations. These measures help align the deployment with the original business objectives and success criteria while acknowledging and managing ethical risks.

6.3 Monitoring plan

For a realistic deployment we propose a monitoring plan that tracks both technical performance and business-level behaviour of the model over time.

On the technical side, we would log model inputs and predictions for all served listings (in an anonymised form where necessary) and periodically recompute evaluation metrics on recent data whenever ground-truth prices become available. At regular intervals (e.g. monthly or quarterly) we would recalculate MAE, RMSE and R^2 on recent transactions and repeat the segment-wise analysis by district and property type, using the same methodology as in our bias analysis. Thresholds can be defined for global metrics and for selected key segments; if these thresholds are exceeded or if data distributions drift significantly, a retraining or recalibration process is triggered.

On the business side, we would monitor how often agents override the suggested price, how large these overrides are, and whether there are systematic patterns by region or property type. This information complements the quantitative error metrics and helps assess whether the system still supports the business objectives of realistic pricing and reduced manual effort.

All monitoring procedures and thresholds should be documented and reviewed periodically. In combination with regular retraining on updated data, this monitoring plan aims to keep the model aligned with the original success criteria and to detect emerging risks early.

6.4 Reproducibility reflection

Throughout the project we documented data preparation, modeling and evaluation steps in a knowledge graph using PROV-O and related vocabularies. This includes the main preprocessing operations, the exact hyper-parameters of the final XGBoost model, the train/validation/test split procedure (with fixed random seeds) and the evaluation metrics used. These records support reproducibility of the experiment by allowing another analyst to reconstruct the pipeline given access to the same raw data and software environment. In addition, the exact versions and dependencies of our environment are fully captured using the Python dependency manager UV. Just with a simple uv sync we can have all the necessary dependencies downloaded in a virtual environment and then we can simply run the notebook without any complications.

However, some limitations remain. The original real-estate dataset may not be publicly accessible in exactly the same form in the future, and external state-of-the-art results (e.g. the Kaggle Random Forest solution) depend on third-party code and preprocessing choices that we do not control.

For deployment, we recommend combining the existing provenance information with explicit versioning of code (e.g. git tags), environment descriptions (e.g. requirements files or container images) and data snapshots. If these elements are maintained together with the knowledge graph, the system can be retrained and audited in a reproducible way when the model is updated or re-evaluated in the future.

7 Conclusion

In this project we built a proof-of-concept price recommendation model for Portuguese real-estate listings. After a series of data preparation steps (dropping columns with many missing values, removing incomplete rows, handling outliers, scaling and log-transforming the target, and one-hot encoding), we trained and

tuned an XGBoost regressor. The final model achieved very strong performance on the held-out test set, clearly outperforming a simple mean baseline and coming close to state-of-the-art results reported on the same dataset. At the same time, our per-segment analysis showed that performance is not uniform: the model works best for common segments such as apartments and houses in well-represented districts, while error rates are noticeably higher for rare property types and small regions.

From a process perspective, we learned how much of a real data-mining project is driven by data quality and preprocessing decisions rather than by the choice of algorithm alone. The impact of handling outliers, choosing an appropriate target transformation (log-price), and designing a robust train/validation/test split with stratification was very visible in the results. We also gained experience with capturing provenance using PROV-O and related vocabularies: every step (data cleaning, model training, evaluation, bias analysis) becomes traceable and, in principle, reproducible. However, we also realized how quickly the provenance graph grows and how important it is to have clear structure and naming conventions, otherwise it becomes difficult to keep an overview. Overall, the exercise reinforced the idea that good documentation and careful evaluation across segments are essential parts of building models that could be used in a realistic business setting.

8 Reproducibility

The workflow is explicitly organized according to CRISP-DM phases (Business Understanding through Deployment), enabling structured reconstruction of decisions and artifacts (data preparation steps, modeling choices, evaluation, and bias checks). The project records the preprocessing and modeling pipeline in a provenance-oriented knowledge graph using PROV-O and related vocabularies, including key activities (cleaning, outlier removal, scaling, encoding), the split procedure, model hyperparameters, and evaluation metrics. This supports auditability and reconstruction provided the same data and software environment are available.

We ensured that our work can be reproduced by making the full project available in a public GitHub repository (until the end of February 2026) and releasing it under the MIT License (GitHub Repo). We created a GitHub Release because we could not link our repository to Zenodo and get a DOI for it. On TU Wien GitLab it was also not possible to create a new project and we haven't gotten permission is time. The GitHub Release link:GitHub Final1 Release. We used the Kaggle dataset [1] as the single, documented data source, and we referenced it directly in the repository so others can obtain the exact same input data. To make the software environment consistent across machines, we managed and locked our Python dependencies with the uv dependency manager. Finally, we provided clear run instructions in the repository (including the required commands) so a reader can set up the environment, execute the pipeline end-to-end, and reproduce the results from the same data and codebase.

9 Feedback

We appreciate the intention behind this exercise, especially the focus on reproducibility and provenance, which is unusual but relevant for real-world projects. At the same time, we would like to share some constructive feedback.

For us, the assignment felt very complex and sometimes difficult to navigate. A large portion of our time went into writing and updating PROV-O triples and making sure all activities and comments were correctly linked, often in the middle of the coding workflow. This made it hard to keep a clear overview of the actual CRISP-DM steps and to see the "big picture" of the process. We often had the impression that we were working more on bookkeeping than on the analytical or business-intelligence aspects of the problem.

Related to that, we felt that the core elements of business intelligence and data mining – such as exploratory visualizations, more varied modeling, and interpretation of patterns – were relatively under-emphasised compared to the provenance documentation. The exercise took a substantial amount of time, but we do not feel that we learned as much about BI concepts as we might have in a more analysis-focused assignment.

If the exercise is used again, we would kindly suggest simplifying the provenance part, for example by providing more ready-made templates or automating more of the triple generation, reducing the amount of manual documentation required, and putting a bit more emphasis on exploratory analysis and interpretation (plots, dashboards, business discussion) so that the link to "business intelligence" is clearer. We hope this feedback is helpful; it is meant respectfully, and we appreciate the effort that went into designing the assignment.

References

[1] Thomas Gaehtgens, Otto Mark, and Fong Andrew. Real estate listings in portugal (portugal real estate 2024). Kaggle dataset, 2024. Accessed: 2025-12-10, MIT License, Original copyright: (c) 2013 Mark Otto; (c) 2017 Andrew Fong.