SAPIENZA
UNIVERSITÀ DI ROMA

DEPARTMENT OF COMPUTER, CONTROL, AND MANAGEMENT
ENGINEERING ANTONIO RUBERTI

Comparative Study of Classification Algorithms on Rome Rent Prices Dataset

MACHINE LEARNING

Instructor:
Federico Fusco
Fabio Patrizi

Students:
Antonio Turco
1986183
Damiano Spadaccini
1986173

Academic Year 2025/2026

Contents

1 Introduction

1.1 Motivation

1.2 Objectives

The goal of this project is to develop an understanding of foundational supervised learning algorithms by implementing, analyzing, and comparing multiple models on a real-world dataset. Specifically, we aim to:

- Implement and train various classification algorithms
- Evaluate model performance using multiple metrics
- Compare and contrast different approaches
- Analyze the trade-offs between model complexity and performance

1.3 Report Structure

During this assignment, the SEMMA methodology will be followed:

> According to (Moine et al., 2011), the SEMMA methodology has five basic phases: Sample, Explore, Modify, Model and Assess. From (Olson and Delen, 2008) SEMMA facilitates the statistical exploration, the visualization techniques and the selection and transforming of the relevant variables in prediction, also can model the variables for prediction processes and later validate the precision of the model.

This report is organized as follows:

- Section 2: Description of the dataset and preprocessing steps
- Section 3: Methodology and model descriptions
- Section 4: Experimental results and evaluation
- Section 5: Comparative analysis and discussion
- Section 6: Conclusions and future work

2 Dataset Description

2.1 Dataset Selection

The dataset is taken from Kaggle and was uploaded 2 years ago by user Tommaso Ramella. It contains data scraped from the website Immobiliare.it about housing announcements in Italy in the year 2023. The author provides the public with two different datasets: the first is about rentals and the second contains purchase listings. Two different versions of each are available, a raw one, consisting of the original data, and a clean one. On the page, the author states the intention to provide a notebook with the methodology applied to parse and clean the data, but we were not able to find the complete version of this notebook (a part of the process can be found in the data publisher's GitHub folder) and as such we assume it was never released. While at the start we were inclined to just use the clean version for rents, in the end we decided against it and tried to gain more insight into the data by doing the work ourselves and trying to get as many features as we could.

The raw dataset contains around 126000 entries. Due to the number of elements in the original set, it was decided to apply machine learning techniques to a smaller subset of interest; that is, the rents in the city of Rome, a sore spot for students and a field with very practical ramifications and consequences. Our objective was to create a way for someone to get a basic understanding of the price a house should have given its characteristics, to avoid being fooled.

The records in the dataset related to Rome (`rome_rents_raw.csv`) are 13276, with 38 columns of information for each of them.

2.2 Data Preprocessing

This section outlines the preprocessing pipeline implemented to transform raw rental listings into a structured dataset suitable for analysis and modeling.

The first step in the preprocessing pipeline involves standardizing the dataset structure. All column headers in the raw dataset, originally in Italian, are renamed to English to ensure consistency and readability throughout the analysis. Following this translation, specific cleaning operations are performed on each column to handle missing values and remove noise.

2.2.1 Handling Missing or Noisy Data

The raw data presented several challenges typical of web-scraped real estate listings. Numeric fields contained currency symbols, unit suffixes, and text values that needed systematic cleaning. For monetary columns like `price`, `condo_fees`, and `deposit`, we removed currency symbols (€) and text indicators (e.g., "/mese"), converting text

representations of zero ("nessuna", "n.d.") to 0.0 and parsing valid entries as floats. Similarly, `square_meters` required stripping the m^2 suffix before numeric conversion.

Room and bathroom columns used notations like "5+" so we addressed this by extracting the base integer while creating separate boolean flags (`more_than_5_rooms`, `more_than_3_bathrooms`) to preserve the additional information without compromising data type integrity. The `floor` column required more sophisticated normalization, mapping Italian descriptions (ex. ground floor, basement ec...) to standardized integers (0 for ground floor, -1 for basement). We had this approach based on the hypothesis that a higher floor number could correlate with higher rental prices in Rome, due to better views and reduced street noise.

2.2.2 Complex Feature Extraction

Several columns column contained structured information embedded within seemingly unstructured text, requiring sophisticated parsing to extract multiple features from single fields.

The `floor` column exemplifies this complexity. Beyond the floor number itself that we already addressed, descriptions often included accessibility information such as elevator availability or disability access. We extracted some distinct boolean features from this field: elevator presence, disability accessibility, raised floor status, multi-floor properties and top floor location.

The `contract` field proved particularly rich, as Italian rental law recognizes multiple contract types with varying legal implications. We parsed nine features including contract type, minimum duration and renewal terms (often expressed as "3+2" for 3 years plus 2-year renewal), and specialized categories such as student rentals, transitional contracts, and buyout options.

From `rooms_details`, we separated total room count from bedrooms and other spaces, classified kitchen types (reflecting Italian distinctions between full kitchens and kitchenettes), and identified special amenities. The `condition` field was split into overall property condition and renovation status, later one-hot encoded into binary features.

Parking information (`parking_spaces`) was decomposed into counts for garage/box spaces, outdoor parking, common areas, and private boxes, with corresponding boolean flags for presence detection.

The `other_features` column yielded 31 binary indicators for amenities ranging from outdoor spaces (terrace, balconies, garden, swimming pool) to security features (alarm, video intercom, security door), utilities (fiber optic, elevator), furnishing status, and property orientation (south-facing, east-facing, internal/external exposure).

Energy efficiency was handled by extracting both the categorical class (A–G, mapped to numeric values 1–7) and the consumption value in $kWh/m^2 \cdot year$ using regex patterns to accommodate various formats. The `availability` field was simplified to a boolean

indicating immediate availability, while `year_built` was cleaned and converted to an integer `construction_year`.

Finally, the free-text `description` column was converted to structured boolean features through keyword extraction. We identified high-value indicators of location amenities (subway, train station, university, hospital, park), property characteristics (bright, quiet), fixtures (shower, bathtub, wardrobe), and nearby services (market), transforming unstructured narrative into quantifiable features that preserve semantic information about desirability and accessibility.

2.2.3 Adding Geographic Information

Given that listings provided only neighborhood names, we enriched the dataset with geographic context by mapping each neighborhood to three features using an external reference file: a boolean indicator for location within the GRA (Great Ring Road, `inside_gra`), a categorical zone feature (`zone`: center, north, south, east, west), and the administrative municipality (`municipality`). This geographic stratification provides essential location context for price prediction, as rental values in Rome vary significantly by area.

2.2.4 One-Hot Encoding

We transformed categorical columns into comprehensive boolean feature sets to capture property characteristics systematically.

The `property_type` column generated 22 binary flags covering dwelling types (condominium, various villa configurations, penthouse, loft, farmhouse), luxury classifications (prestigious, middle, economical class), and ownership structures (full, partial, usufruct, timeshare).

6 Conclusions

6.1 Summary of Findings

6.2 Key Takeaways

6.3 Limitations

6.4 Future Work