Machine Learning Challenge

Barbara Karwowska 3178784, Kaggle: Barbara Karwowska

May 9, 2023

1 Introduction

The overall goal of my work is to train a model for predicting house prices in different Italian cities based on the data set containing data on their selected characteristics.

2 Data preprocessing and feature engineering

As the first step, I do some preliminary data exploration to get an understanding of information available and devise a gradual strategy for cleaning the data and building the model. I look at statistical properties of the target variable ”price” and notice that its distribution is positively skewed with a large standard deviation and a long tail due to extreme outliers. To reduce the impact of extreme values and approximately normalize the distribution, I apply the logarithmic transformation to price data and consider ”price_log” as the new target variable for the modelling purposes.
Before analyzing relationships between features, I encode the categorical variables and convert their data type to numeric to be able to include them in the correlation matrix from the beginning. For ”conditions”, I use OrdinalEncoder where I specify the order of categories so that 0 refers to the worst (Da ristrutturare) and 3 to the best condition (Nuovo / Nuova costruzione). For ”balcony” and ”garden”, I encode True with 1 and False with 0.
Now I compute the correlation matrix and represent it with the heat map to see the degrees of correlation between the features. This is helpful for determining the way of handling missing observations as well as the overall strategy for modelling the dependency between the price and house characteristics.
From the heat map, one can observe that latitude and longitude are strongly negatively correlated, which turns out to be due to the particular nature of the sample of cities in which the houses in the data set are located (which I found to be Milan, Rome and Venice). This relation is also the consequence of Italy’s shape as a country. The seemingly perfect correlation between the elevator and the construction year, total number of floors and expenses is the result of limited data (when elevator is 0, the data corresponding to these 3 variables is missing, hence such a relation). Based on the heat map, one can however observe that there is an important relationship between three features: surface, number of rooms and number of bathrooms. I will look into it more closely while filling in the missing observations.
Now I inspect the variables which, in absolute terms, are the most correlated with the price. They are: ”n_bathrooms”, ”surface”, ”n_rooms”, ”construction_year”,”longitude”, ”latitude”, ”floor”, ”proximity_to_center”, ”conditions”. Overall, the correlation levels between these variables and the price are not very significant. This may be pointed to the fact that these relationships are most likely not linear, but also that there is a range of forces pushing in opposite directions, while jointly shaping the price (for example, although the proximity to the center

normally drives the price up, it also tends to coincide with the smaller surface area, which pushes the price down). We did not display balcony and garden on the heat map due to missing data (only values True are present, otherwise there is nan), but we expect these variables to have a positive relationship with the price.

I perform feature engineering in order to extract potentially useful information from the variables "latitude" and "longitude": the name of the city in which the house is located as well as the corresponding population density. To do this, I use 2 data sets: one with city names and geographical coordinates and another one with city names and respective population densities. The new variable "city" is aimed to capture the city-specific characteristics affecting the price. To check if they are relevant, I examined and compared the price distributions for these 3 cities. Seeing that they vary, I decided to proceed with OneHotEncoder to numerically encode cities as dummies and to use them further as features. The second new variable, which we consider including, is the population density of the city. The reasoning behind this is that the population density reflects to some extent (with some limitations) the demand for houses in a specific city. However, we decide to abandon this approach, having discovered that the sample of cities is too limited for this variable to have informative value for our model (since there are only 3 cities present in the data set, not being representative enough).
To create a column "city" and fill it with data (name of the city which is the closest to the location of the house in terms of longitude and latitude), we will use the haversine formula (relevant for computing distances between points on a sphere given their longitudes and latitudes). For each house in our data frame, we will calculate the distance between the house location and all cities in the portion of the data frame mapping cities to geographical coordinates. Having tried using considering a larger number of cities, I decided to restrict myself to the cities with population of at least 150,000, the approach justified by the analysis of the clustering patterns. Based on the values of the proximity to center and the overall pattern exhibited by observations, we concluded that the only cities in this data set are Venice, Rome and Milan.
Moreover, we also introduce another feature "age" to represent the age of the building, computed based on construction_year. It is better to use it due to scale more aligned with other features.

3 Missing observations

We create a data frame with percentage of missing values for given variables to devise a strategy for handling them. For "garden", $\frac{2}{3}$ of the observations are missing. The non-missing values are True, so given that we consider cities, it is reasonable to assume that these $\frac{2}{3}$ can be filled with False. For "balcony", we also have the only non-missing observations True. We will assume that the missing ones are False.
For the geographical data: proximity_to_center, latitude, longitude), they are jointly missing for a small proportion of observations, 13, which we drop.
When it comes to another category of data - related to spatial characteristics of a house: surface area, number of rooms and number of bathrooms, I decide to drop the observations for which the surface or the number of rooms is missing. For both variables, less than 1% of observations have missing values, so we can drop them without losing much information. For the number of bathrooms, we will try to estimate the missing values in the training set given the values of surface and the number of rooms. This way we will exploit the correlation between these variables. Before we learn a model for estimating the number of bathrooms, "model_bathrooms", we will discard some outliers from the training set to get the model that generalizes well. First, we do univariate analysis, using .describe() methods for columns interest to spot any anomalies. We observe that the minimum surface is 0, which is not possible for a house. We decide to set

a threshold of 15 and discard observations with surface below it. The maximum surface in the data set is likely, so we do not set any upper bound. For the number of rooms and the number of bathrooms, we cannot say anything conclusive by considering them separately. Hence, taking account of their mutual dependence, we consider the outliers by looking jointly at the variables. We plot the number of bathrooms against the surface and the number of bathrooms against the number of rooms and drop several outlying observations. By visual inspection of the graph, we decide to drop the observations for which the number of bathrooms is 5 or 6 while the surface is below 150 and when n_bathrooms is 8 or 9 and the surface below 200. We also decide to drop the several observations for which the number of bathrooms exceeds the number of rooms, considering them as outliers. I apply the logarithmic transformation to the number of bathrooms given that the distribution of this variable is right skewed. Moreover, I scale both y_log and explanatory variables using standard scalers to account for the difference in scale. I train train different models on the data, looking for the one minimizing the mean squared error. Given the correlation between variables, using k-fold cross validation technique to improve the performance and limit overfitting, I first trained LinearRegression with polynomial features (of different degrees, but eventually degree of 2 worked the best). I also decided to consider the random forest and run the hyperparameter tuning algorithm. Best hyperparameters were given as: n_estimators: 400, max_depth: 10. I use them to train the random forest on our data. In linear regression, the MSE was 0.6882581874427428 and for the random forest MSE was 0.4483313782022949 (both on logarithmically transformed and rescaled target variables and on rescaled features). Comparing these values, we proceed with random forest and assign it to "model_bathrooms". Using this model, I predict missing values for n_bathrooms in the training set and fill them in.
In this part, except for model_bathrooms, I learn also 2 other models (which I call model_surface and model_n_rooms) for the later use. They are aimed to fill the missing observations of the number of rooms and the surface in test data, but need to be trained on the training data to avoid overfitting.
I decide to drop 3 variables with many missing observations: total_floors, energy_efficiency and expenses. These variables have huge standard deviation, so there is a big uncertainty associated with their values. Moreover, from the analysis of the correlation matrix as a heat map, there was no significant correlation with other variables, which makes it even harder to estimate them. Moreover, since they have a large proportion of observations missing, their informative value decreases even more. Also, since they are not strongly correlated with the price, the potential costs of dropping them are relatively low. Also empirically, these characteristics do not seem to have a strong causal relation with the price. Due to limited correlation levels with other variables, for the remaining variables with missing observations, I use SimpleImputer, specifying mode strategy for elevator (binary variable), mean for construction_year (approximately normally distributed) and median for floor (not a regular distribution). For conditions, I drop the missing observations, since they do not constitute a high proportion of all data points.

4 Outliers

I treat outliers referring to real-world data. I have already given a treatment to outliers relative to the surface, n_rooms and n_bathrooms. Now, I tackle the remaining variables.
For construction year, I discard observations with values above 2023, since they are not plausible and must be a result of error. We also discard the values below 1800 given their low likelihood, concluded based on empirical data. For floor, we discard values below -1 and above 20 considering them as either extremely rare or even impossible. We leave other variables as they are, as we did not spot any similar anomaly.

5 Model

I standard scalers to rescale both features and target variable, to account for differences in scale which could affect the performance of the models. I take price_log to be the target variable. To learn the model, I decide to include following features: balcony, conditions, floor, garden, n_bathrooms, elevator, n_rooms, proximity_to_center, surface, age, city_Milan, city_Rome, city_Venice. Doing that, I decide to abandon latitude and longitude since the information is already contained in variables proximity_to_center, city_Milan, city_Rome, city_Venice. I also decide to abandon construction year, using age instead, due to a more convenient scale. I also dropped expenses, total floors and energy efficiency. I also abandon population density given the insufficient sample of cities. I try several models with different parameters and select the one minimizing MSE. Before using each techniques, I use hyperparameter tuning algorithm to pick the optimal model from a given class. I used following models with following hyperparameters and values of MSE:
1. Elastic net with best alpha: $1e-05$ and best l1_ratio: 1.0 and MSE: 0.43.
2. Ridge regression: with alpha of 10 and MSE of 0.43.
3. Lasso with alpha of 0.001 and MSE of 0.43.
4. Random forest with 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 800 and MSE of 0.29.
5. SVM with 'C': 1, 'epsilon': 0.1, 'kernel': 'rbf', and MSE of 0.32.
6. Neural network with MSE of 0.31, but I consider the model too complex for such a small data set and recognize the risk of overfitting.
7. Gradient boosting with 'C': 1, 'epsilon': 0.1, 'kernel': 'rbf' and MSE of 0.29.
Given the findings, we proceed with random forest with specified parameters to predict the house prices.

6 Test data

We treat the test data the same way as the training data, where possible. Hence, we drop energy efficiency, expenses and total floors, we encode categorical variables and add explanatory variables the way we did before. We use onehotencoder for cities. We drop observations with the construction year above 2023 or below 1800 and age below 0 or above 223. Since for floor, we cannot drop observations, we replace values below -1 with -1 and above 20 with 20. Similarly, we replace surface below 15 with 15. For the spatial characteristics of the house we are also taking a particular approach. For houses with number of bathrooms above the number of rooms, we replace the value of the former with the value of the latter. For houses with number of bathrooms 5 or 6 and surface below 150 or number of bathrooms above 7 and surface below 200, we replace the value of n_bathrooms with nan. We will estimate it using model_bathrooms, we learnt before. First, we fill in the missing values for the surface and n_rooms. We use model_surface and model_n_rooms learnt on the training data. We can do that for all observations except for 5, for which both values for n_rooms and surface are missing. For these 5 observations, we use mean imputations strategy for the surface (since its distribution is approximately normal) and median for n_rooms. Now we fill in the missing values for n_bathrooms using the model learnt before model_bathrooms. For Floor,construction year, elevator, we use the same imputation strategies as before: median, mean and mode. For conditions, we will inpute median and for city mode. For proximity to the center we consider the median of this variable for the chosen city.