Predicting House Prices

Davide Romano Student ID:3164081

Università Commerciale Luigi Bocconi

Data-Exploration

In my analysis, I started with data-exploration to better understand the information contained in the "train.csv" dataset. Data-exploration is a key step in data analysis, as it allows us to understand the distribution of variables, the presence of missing values, and the correlation between different features.

I used the "info" function to obtain information about the columns in the dataset, particularly the type of data and the presence of null values.

To get an overview of the distribution of the numeric variables, I used the "hist" function of the Matplotlib library to plot a histogram of each numeric feature in the dataset. In this way I was able to see the distribution of the features and identify any outliers.

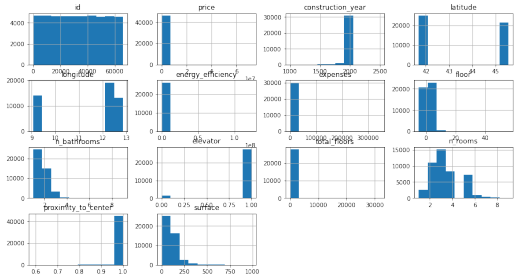

Figure 1: *histrogram of the numeric features*

I then used the "heatmap" function of the Seaborn library to create a heatmap of the correlations between the numerical features in the dataset. In this way, I was able to visualize the strength of the correlations between the different features and identify any relationships between them.

Finally, I used the "isna" function to count the missing values within the dataset. Specifically, I used the "sum" function to sum the missing values for each column to get an idea of the amount of missing data in the dataset.
The columns of the train dataset have up to 31000 missing values, that must be managed in some way, since I cannot run the regression until I have 0 NaN in the dataset.

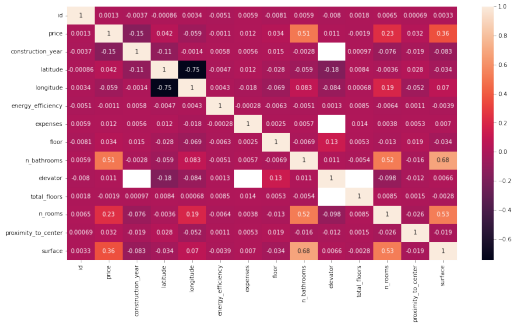

Figure 2: *heatmap*

Pre-Processing

For the pre-processing phase, I started by eliminating the 'id' column since it is not useful for house price prediction. Next, I addressed the issue of missing values in the dataset variables. I decided to replace these missing values with the median or the mode depending on the variable. I decided to use this method because the mean could be affected by the presence of outliers and could lead to a skewed result.

I replaced the missing values in the variables 'n rooms,' 'floor,' 'n bathrooms,' 'surface,' 'construction year,' 'energy efficiency,' 'expenses,' 'proximity to center,' 'latitude,' 'longitude,' and 'total floors' with the median. For the boolean variables 'garden', 'balcony' and 'elevator', I replaced the missing values with the mode, which represents the most common value in the variable.

In addition, I converted the boolean variables to integers to simplify the modeling and make the calculations more efficient.

Finally, I checked again for missing values in the dataset using the method isna:
All the variables have now 0 NaN, except for 'conditions', that 1229 NaN. The problem is that this is a cathegorical variable, so I cannot substitute the NaN. I now have to transform this variable into a numerical one.

```
price                    0
balcony                  0
conditions            1229
construction_year        0
latitude                 0
longitude                0
energy_efficiency        0
expenses                 0
floor                    0
garden                   0
n_bathrooms              0
elevator                 0
total_floors             0
n_rooms                  0
proximity_to_center      0
surface                  0
dtype: int64
```

Figure 3: *Missing value check*

Outliers

I then dealt with data outliers, that is, extreme values that could skew the distribution of the data or even compromise the validity of the analysis results.

To do this, I decided to focus on the highest values present in certain variables. For example, I noticed that the highest building has as many as 31,906 floors, while the second highest has 135, so I decided to eliminate the first one. I also eliminated the lowest value of the variable "price" and the highest value of the variable "construction year" (since it corresponds to a very future year, 2500).

To do this, I sorted the data in ascending order and selected the highest or lowest value, and then selected all data below the highest value or above the lowest value, thus eliminating outliers.

I decided not to use the IQR method because I would have eliminated too much data, consequently making the model less accurate.

Point of Interest

I then moved on to cleaning and processing the "poi.csv" dataset. First, the dataset is imported through the pandas library and a copy is created to avoid modifying the original dataset. I then selected only a few columns deemed useful as features (characteristics) for the regression model, such as latitude and longitude.

Next, the dataset is further processed through the creation of new variables indicating the presence or absence of certain POI categories. For example, the variable "culture" indicates the presence of theaters, universities, libraries, schools, museums or art galleries. The variable "health" indicates the presence of pharmacies, doctors, dentists, hospitals or clinics. The variable "entertainment" indicates the presence of restaurants, pubs or cafes. Finally, the "value" variable indicates the presence of points of interest that might add value to an area such as cinemas, fountains, banks, social clubs, music schools, or parking lots.
Finally, further cleaning of the dataset is performed, removing rows where none of the new variables indicated are present. This allows for only those POIs that are useful as features for the regression model.

Feature Engineering

At this point I went to solve the problems I was having with the categorical variable 'conditions'. Specifically, I created dummy variables. To do this, I first identified all the unique values in that variable except the last one (which corresponds to when the variable is missing), and then I used pandas' "get dummies" function to transform the variable into a series of binary columns, one for each unique value. In this way I got a set of new variables representing the presence or absence of each possible condition for each observation. After creating the dummy variables, I deleted the original "conditions" variable.

I then tried to create new variables that could be useful in predicting the price of the property. In particular, I created the variables "floor relative," "garden with surface," and "surface per rooms."

The first variable, "floor relative," represents the proportion between the floor of the apartment and the total number of floors in the building. I thought this proportion might be important, since apartments on higher floors might have a better view and thus a higher value.

The second variable, "garden with surface," represents the area of the garden divided by the total area of the building. This variable takes into account the fact that the presence of a garden could increase the value of the property.

Finally, the third variable, "surface per rooms," represents the surface area of the property divided

by the number of rooms. I thought this variable might be useful since larger apartments (with more surface area) might have a higher value.

To create these variables, I used simple mathematical operations among the variables in the dataset. After creating the new variables, I analyzed the correlation between them and the price of the property using the heatmap and verified that these new variables had a fairly significant correlation with the price.

I also tried to write a function that could determine whether a house was close to one or more of the points of interest I had selected from the 'poi' dataset, but it was too computationally complex and caused the system to crash.

Figure 4: *heatmap*

Regression

I ran a regression using the Random Forest algorithm.The Random Forest has several advantages over other machine learning algorithms such as resistance to noise and the ability to handle large datasets. In addition, the output of the Random Forest also includes the significance of individual features which allows selecting the most significant features for target prediction.

I divided the dataset into a training set and a test set, using 33 percent of the data for testing. Next, I trained the model on the training data and made predictions on the test data.

I used the parameter max depth=2, which controls the maximum depth of the tree, and random state=42 to ensure the reproducibility of the result.

Finally, I calculated the mean square error (MSE) between the predictions and the actual values and created a table of feature importance, which indicates the importance of the variables for the purpose of predicting the property price.

Because I removed very few outliers, the MSE of this model is affected, but by removing so little data I did not take away information from the model. This will ensure that I can get a better result in a dataset with fewer outliers.

Grid Search

Grid search is a method for finding the best hyperparameters of a machine learning model. The process of selecting hyperparameters is important because it affects the model's ability to generalize to new data.

The process of manually searching for hyperparameters can be laborious and does not always guarantee the selection of the optimal set of hyperparameters.

To solve this problem, the Grid Search explores a set of possible combinations of hyperparameters, trains and evaluates the model on each of them, returning the one that gets the best predetermined score based on a chosen evaluation metric. In this way, Grid Search can automate the hyperparameter search process and improve the performance of the machine learning model.

In my case, I compared the MSE obtained by the previous regression with that obtained by Grid Search. The latter obtained a lower MSE.

PCA

Next, I applied PCA (Principal Component Analysis), which is a dimensionality reduction technique to detect the principal linear combinations of variables that explain most of the variance in the data. The goal is to transform a set of correlated variables into a new set of linearly independent variables called principal components. In this way, the dimensionality of the dataset can be reduced by removing the less informative variables and keeping only the most significant ones. In the code, the data are scaled and then PCA is applied to find the number of components that explain at least 95 percent of the variance of the dataset. This is useful because it reduces the number of variables and simplifies the model, reducing the risk of overfitting and improving the generalization of the model.

Conclusion

In conclusion, the error I obtained by applying the model to the test set indicates that it is definitely improvable, for example by adding more feature variables that could add interesting information. Having removed few outliers the error on the train test is very large, but by doing so I avoided having an inconclusive model.