Report of Applied Predictive Modelling
Home sales prices

Davide Ratto
Matricola 827130

***Abstract*—The real estate market is one of the most classic examples of the application of statistical learning techniques. The variable of interest is usually the price of the house, which allows real estate marketers to have good estimates of the evolution of the prices of immmovable properties depending on the change of different parameters such as neighborhood, number of rooms, condition of the house and the size of the property. In this project, classical forecasting models but also models more related to the sphere of machine learning were used, with the aim of predicting the sales price of different real estate properties sold between 2014 and 2015.**

I. Introduction

This report describes the basic steps of carrying out the analysis done on the Home sales prices dataset.

One has a Train on which the evaluation of different models will have to be performed, with which one will then return the predictions of the variable $Price$ for the Test set at hand. The metric of interest that will be attempted to minimize is the Mean Absolute Error (MAE).

It was decided to split the Train set to also obtain a Validation set on which to go to evaluate the best model.

An exploratory analysis was conducted aimed at identifying possible problems in the variables, then the different problems were solved through feature engineering, which is intended to then allow the models to be able to use the most informative and well-specified covariates possible.

In the modeling part, several models were trained including Linear Regression, Random Forest and XGBoost, then evaluated on the Validation set left aside before modeling, with the purpose of getting the metrics as general, and therefore comparable, as possible.

An ensemble of the different methods was then tried with the aim of minimizing the metric of interest.

Finally, the best model obtained on the complete Train set was reestimated and $Price$ predictions were returned for the Test set.

II. The dataset

The Home sales prices dataset contains real estate data for 21613 properties and is divided, as mentioned earlier, into Train and Test.

The Train consists of 17293 statistical units with which the target variable $Price$ is associated, already transformed with a logarithm to base 10. The Test consists of the remaining 4320 units whose price is to be predicted.

There are 18 variables in our possession and they can be divided into three macrocategories:

- Variables referred to the property;
- Variables referred to the location of the property;
- Variables referred to neighboring properties.

III. Exploratory analysis

What jumps out at an initial analysis is that several variables are poorly coded. For example, the variable $bedrooms$, which refers to the number of rooms, is of numeric type when it is clearly a factor variable. Like the latter there are others that will therefore be converted from numeric to factor.

An important step in the exploratory is to check for collinearity, that is, to identify overly correlated variables. It was chosen to remove above a threshold of 80%. The variables that crossed the threshold were $sqft\_above$ and yr_build, so they will not be used in modeling.

Analyzing the summaries shows that certain levels of the factor $bedrooms$, $bathrooms$ and $floors$ are sparsely populated. These will later be merged to obtain more numerous and thus more informative levels.

In $bedrooms$ there is a row with 33 bedrooms, probably a typo. This has been removed.

Another problem given by the levels of some variables is a logical one. In fact $bathrooms$ and $floors$ have decimal values such as 3.5, which is meaningless since the number of bathrooms is discrete.

IV. Data pre-processing

Regarding the problem on the levels of the variables $bathrooms$ and $floors$, it was decided, regarding the modes that will not be merged due to low populousness, to keep these anomalous modes as they are, without grouping them with the modes that are more similar to them.

For example, for statistical units with $bathrooms$ equal to 3.5, a grouping with those of level 3 could be assumed. This elaboration was initially done, but it was realized that too much information was being lost and therefore the performance of the models was getting worse.

Although there are nonsense levels, these contain information about the dependent variable that proves useful, since there seems to be a real difference in price between statistical units with an anomalous mode and statistical units with which, by modes close in terms of ordering (we are talking about ordinal qualitatives), they should be grouped.

Below are the pre-processing done for each variable:

- $Bedrooms$: you grouped the underpopulated levels '0' and '1' into a new level '$\leq$1' and levels '7', '8', '9',

’10’, ’11’ into the new level ’>7’;

- $Bathrooms$: the sparsely populated ’0’, ’0.5’, ’0.75’ levels are grouped into a single ’<1’ level; the ’1.25’ level is merged with the already highly populated ’1’ level. From ’4.5’ up to ’8’ a single level called ’5’ is created;
- $Floors$: level ’3.5’ is sparsely populated and is merged with level ’3’.

$Zipcode$ is a feature indicating the neighborhood in which the property is located. Since it consists of 70 levels, also very similar to each other in terms of the average values of the variable $Price$ conditioned on them, it was initially thought to merge them to reduce the number and thus to better compress the information contained. However, following modeling tests with the new levels, it was concluded that it is much more informative and useful for forecasting purposes to leave the levels of this variable unchanged.

V. Feature Engineering

The variable $date\_sold$ can be better specified, in fact in two short steps we went from the yyyy/mm/dd format to having three different variables for year, month, and day.

It was found useful to construct a new variable using yr_built and $year$, which indicate the year the property was built and the year it was renovated, respectively (if the house was not renovated it takes a value equal to the year it was built).

The new feature will be constructed in two steps:

- The difference between renewal year and construction year is made. The following discrete variable in relation to price is obtained, as shown in Figure 1:

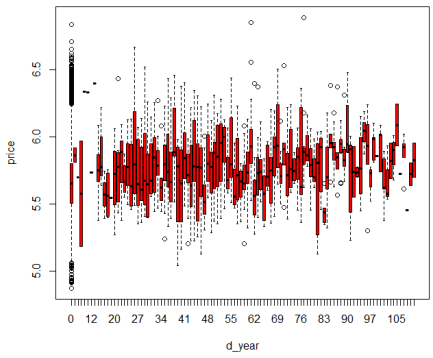


Figure 1.

- The values obtained are further merged to obtain more populated classes. It seems reasonable to create 6 groups, one of which will be the never-restored, coded with ’0’. The others will be created by grouping from ’0-20’, ’20-40’, ’40-60’, ’60-80’, ’80-100’ and ’100-114’.
  Here, in Figure 2, is what the boxplot looks like at the end of the process:

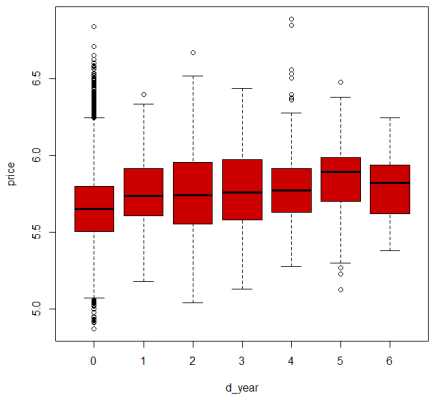


Figure 2.

Since there are very similar levels for conditional $price$ statistics, we merge together modes ’1’ to ’4’ and ’5’ to ’6’.

VI. Data Modeling

Performance will be evaluated on a Validation set, which will allow comparison of models on the same dataset without the latter entering the modeling process. The split in train and test will be of the 80/20 type.

After performing the split, diagnostics of the outliers on the new train were performed. The method of cook distances was used.

From Figure 3 we can see the presence of several points above the threshold (red line), which usually for cook distances is set at 4 times the average of the same distances.

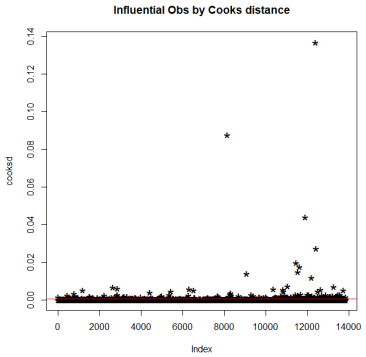


Figure 3.

It was decided to remove only the outliers with cook $>$ distance of 0.035, since it was deemed useful to keep the other observations because they are presumptively representative of the luxury sector.

Logically, when we go to train the best model on the whole Train dataset, and then go to make predictions on the real Test set, we will redo the analysis of the outliers, again removing only the excessively large ones.

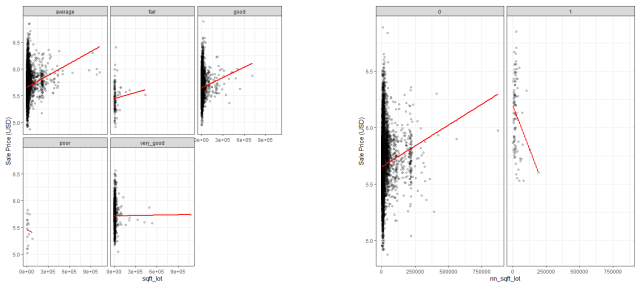
The models are built and trained through the tidymodels framework, which allows for more control and flexibility in building the models.

A. Linear Regression

Several lineri regressions were tried, using initial covariates, adding second-degree polynomials, inserting different interaction types and various splines.

The variable $Year$, deemed uninformative, was removed from the model covariates. Other variables, which initially seemed uninformative, were nonetheless found to be useful for prediction purposes.

Interactions were chosen based on graphical feedback of which, in Figure 4, two examples of the multiple interactions identified are shown.



(a) sqft_lot with condition (b) nn_sqft_lot with waterfront

Figure 4.

After several tests, the one containing the second-degree polynomials of the variables ^{sqft}, $sqft\_living$, $sqft\_lot$, $lattitude$, $longitude$, nn_sqft_living, nn_sqft_lot, the splines of degree 20 of $lattitude$, $longitude$, $sqft\_lot$ and 8 interactions of the continuous variables especially with $zipcode$ and $floors$.

The plot of residuals was checked and no outliers were found.

B. Random forest

Random Forest was tuned through a 5-fold cross-validation using the grid search method. It was tuned for the number of features to be used randomly for each split (mtry), the minimum number of observations a node can contain to be further split (min), and the regularization factor.

It was tried first with 100 and later with 200 trees, noting little improvement in performance, so the number of trees was not further increased.

Both using basic covariates and adding splines were tried. Adding the interactions was not deemed useful since, by construction of the algorithm, Random Forest already creates interactions by itself.

By adding the splines, the best cross-validated MAE of 0.0551 was achieved. The best parameters were found to be:

- mtry = 12;
- min_n = 5;
- regularization = 0.005.

C. XGBoost

As with Random Forest, XGBoost was tuned through the grid search method with a 5-fold cross-validation. Instead, the number of trees was kept fixed at 1000.

The parameters of interest here were the maximum depth a tree can reach (tree_depth), the minimum number of observations a node can contain to be further split (min) and the learning rate.

The initial covariates were used, without any addition of interactions, splines and polynomials. The best model, with a cross-validated MAE of 0.0505, was found to be the following:

- tree_depth = 10;
- min_n = 10;
- learning rate = 0.02.

D. Ensemble

An attempt was made to construct an ensemble among the three models, giving more weight to those with better performance.

The best ensemble was between XGBoost and Linear Regression, assigning 0.75 and 0.25 as weights, respectively.

VII. Risultati

Linear Regression was already initially estimated on the Validation set, while for the other models we had the metrics cross-validated, so we re-estimated them on the Train set and derived the metrics on the Validation set.

Table 1 shows the best results for each class of models, referring to the MAE obtained on the Validation set left aside at the beginning of modeling.

Table I

Models	MAE
Linear regression	0.0525
Random forest	0.0564
XGBoost	0.0494
Ensemble	0.0485

VIII. Discussione

The best model turns out to be the ensemble between the Linear Regression model and the XGBoost, whose predictions were weighted 0.25 and 0.75, respectively, as the XGBoost model performed better than the Linear Regression on the Validation set.

Since the ultimate goal of this project is to obtain the best performance on the Test set, it was deemed essential to train the Linear Regression and XGBoost on the whole Train set, to have the parameter estimates as accurate as possible, and then build the ensemble and return the final predictions.

The Random forest still performs well but inferior to the other models, counting also the fact that the dependent is log10 scaled and therefore small deviations may be, in the price then later transformed, significant.

IX. Conclusione

This report outlined the steps that led to the selection of the ensemble model as the best predictor of the $Price$ variable.

Certainly the Data Preprocessing and Feature eingineering parts were essential in obtaining good metrics in the later part of modeling. Implementing different choices in these two sections could have changed the results and perhaps even the hierarchies between the models.

The choice to keep the outliers not excessively large, in terms of cook distance, certainly affected the final results but was made with the aim of making as little error as possible with regard to the prices of luxury homes, which would probably have generated much higher errors.

The tuning part was very computationally heavy, especially with regard to XGBoost, but it brought significant improvements in performance.

In conclusion to this project are returned the predictions on the Test set and the part of the code related to the analysis carried out, where, however, only the steps necessary for obtaining the final predictions will be included.