Final Project: Fundamentals of Data Science

Sergio Ballesteros, ID 1763191. Course 2016/2017

I. Introduction

In this project I have predicted the houses' prices of the test set obtaining a score on *kaggle.com* of 0.12864 with the username *Ballesteros*. The process was cleaning the data, creating new features, modeling with Extreme Gradient Boosting and checking the results with a k-Fold cross validation (KFCV).

II. Requisites

Beside the libraries used in class, it is needed the library *xgboost*, not included in the anaconda evironment, but can be installed with the following command on GNU/Linux:

```
pip install xgboost; conda install libgcc
```

III. Cleaning the data

First I merged in anew dataframe all the features of the training set (no prices) and test set to treat them at the same time. The *Na* values have different meaning for each feature so I consulted the description of the data set from the kaggle website for all of them. In the categorical features I replaced some of the missing values with the string *No*,this means that the house lacks such feature, as is the case of the feature *PoolQC*, the houses with *Na* in that cell do not have a swimming pool or the I replaced them with the mode or the average value of the column depending on the feature. Then I mapped them to numerical values, in fact, when the meaning of the values can be ordered, like ratings, they are mapped also in a numerical order. In the numerical features the missing values are replaced by 0 because the house lacks such feature, for example the houses with missing *LotFrontage* do not have a frontage. Now, I have one dataframe with only numerical values.

IV. Creating new features

Some combinations of the existing features provide better predictions. I created new features and checked if they improved the prediction, the ones that did it were the ones that follow:

- Total size of the two floors of the house together
- Total liveable size
- Whether the house was remodeled or not
- Whether the house was sold the same year it was built

V. Modeling

Now the dataframe is splitted back in a training set and a test set. The model that I used is the Extreme Gradient Boosting (XGB). I chose this techique for several reasons. First, the dependence of the variables with the price can be complex and not necessarily linear, therefore maybe a linear regression can not exploit all the information. Second, the XGB uses decision trees, this means that this machine learning technique works fine even with the presence of outliers and without normalizing the data. In fact, I compared with a 5 fold cross validation the results of normalized features ($mean = 0$ and $STD = 1$) and non-normalized data and the results were similar, but slightly better without normalizing, 0.8969 ± 0.0333 and 0.8974 ± 0.0325 respectively. Also this technique can be easily ran in parallel, in fact with the python package that I used, *xgboost*, I set up the option of using all the threats of the CPU. The last but not least motivation is that I used this technique because I wanted to learn a new one beside the ones covered in class and the XGB has a very good documentation [1].

VI. Checking the results

In order to validate the model, as well as to verify the feature engineering, a 5 fold cross validation of the whole training set was used. There are some recommendations for the XGB parameters [2], but for this dataset I could find better ones using the KFCV. With the best feature engineering and best parameters found the best score was 0.8974 ± 0.0325 (the maximum possible is 1.0). This leads to a score of 0.12864 on kaggle.
In the version of $1763191.py$ that I send you the KFCV is disabled by default for speed reasons. To enable it simply uncomment the last line of the code.

References

[1] Introduction to Boosted Trees http://xgboost.readthedocs.io/en/latest/model.html

[2] Complete Guide to Parameter Tuning in XGBoost https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/