Rafael Fontana Dias

Feb 2021

Capstone Project 3

Housing Prices Advanced Regression Techniques

**Project Overview**

Predicting the residential house price can guide real estate agents during the decision making and pricing process, therefore assistant their customers to find the best selling price for their houses. Industry leaders also use the prediction of the house price for various purposes.

The data in this project is described in detail by Kaggle playground competition:

House Prices: Advanced Regression Techniques [https://www.kaggle.com/c/house-prices-advanced-regression-techniques]

Zillow uses the prediction of the house price to provide guidance to its online seller and buyer. Airbnb leverages the house price prediction to inform their hosts to better pricing their places. The objective of this project is to predict the final price of a house given 79 explanatory variables that describe almost all aspects of a house in Ames, Iowa. This project is from a Kaggle playground competition (De Cock (2011)).

Kaggle is a platform for data science competition. Participants are challenged to build models given the data provided to make predictions and then submit their results to Kaggle. Kaggle usually provides a training dataset and a test dataset and the task of the participants is to build a model based on the training dataset and make predictions on the test dataset. The submitted results will be evaluated by specific evaluation metrics on the test dataset. During the competition, there will be a score calculated based a fraction of the test dataset shown on the Kaggle Leaderboard, which is known as the public score. The final competition score, also called the private score, is based on the complete test dataset.

**Problem Statement**

Prediction of sale prices.

Let’s say that a buyer is interested in purchasing a house. He or she has an estimate of the price of the house and has an offer in mind. The price estimation might have been based on few factors or external sources such as real estate agencies. The problem for the buyer is knowing the exact amount for the purchase price of the house. For a real estate company, which can also pose as a buyer or broker, the problem is to negotiate for the best deal. This dataset has several factors. It becomes crucial to know the levers that drive the price and develop a model to predict them with best accuracy.

The sales price prediction has many useful applications like enable the client to have a better understanding of the prices changes, advantageous position while negotiating, save money on researching for both clients and companies, acknowledge what is the most influential features on the price composition and so on.

**Data Preprocessing**

When performing regression it is always important to aim for normality, that is does our dependent variable follows normal distribution (theoretically it is important demand). Other than that we should also (in general) look for outliers to remove them from data. We can do a joint plot of all of the variables versus the dependent variables and get a hunch where the outlier might be hidden.

I transformed the dependent variable into something that resembles a more normal distribution, as well as corrected skewness for the intendent features. By transforming the features, it helped in prediction especially for the linear-based models.

So what is our next step in data processing? We should note that ML algorithms can only process numerical data (it needs to be encoded in numerical format), hence we ought to labelEncode (if there is hierarchy in the independent variable, i.e. good - 0, better - 1, best - 2) or Dummy encode (0 and 1 vectors). Next we should also think about what variables are actually categorical even though they numerical or strings. For fully numerical values we should also think about skewness (how can we remedy it?). Is there any text that we can extract info from? And finally what about scaling, we need to make our data comparable and robust to outliers.

For example, our first column MSSUbClass should be actually categorical, and not only that but with some hierarchy also (since there is difference whether it is 120 or 20)

**Data Wrangling**

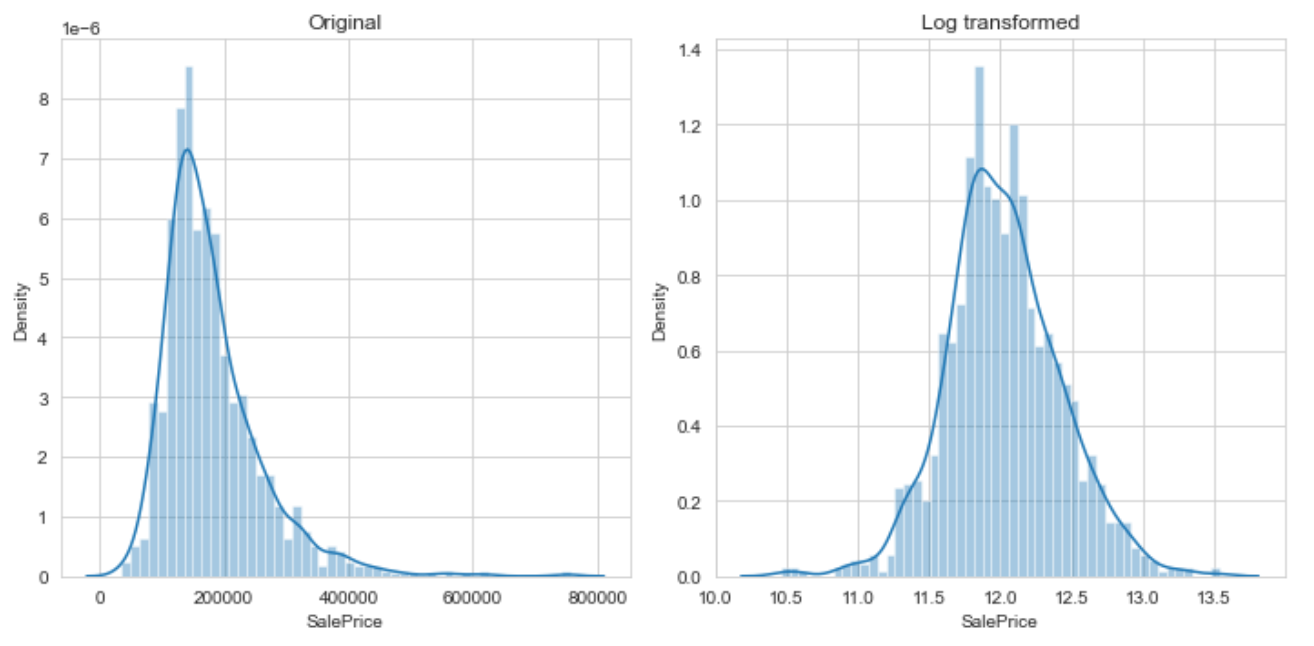
In this section data will be cleaned, variables classified, dealt with missing values and outliers.

**Remove outliers**[**¶**](http://localhost:8889/notebooks/GITHUB/SpringBoard/CapstoneProject_3/ensamble-methods-on-regression.ipynb#Remove-outliers)

We should be careful with outliers, always removing them is not the best choice. It could happen that we have also outliers in the test set. Remove it from training set will not give desired predictions. We should rather opt for another choice and that is scaling and make transformations in order to get our model robust to the outliers. Outliers can be very informative about the subject-area and data collection process. It’s essential to understand how outliers occur and whether they might happen again as a normal part of the process or study area. Unfortunately, resisting the temptation to remove outliers inappropriately can be difficult. Outliers increase the variability in your data, which decreases statistical power. Consequently, excluding outliers can cause your results to become statistically significant

**Data Exploration**

The training dataset is mixed with categorical and numerical features as well as some missing values. Specifically, there are 1460 observations, each with 79 features including the sale price of the house. The test dataset has 1461 observations, each with 78 features. Our task is to fill the sale price of the test dataset using the model trained from the training dataset. The histogram of the house price is shown in Figure 1, which shows that it is skewed. Thus, the log transformation needs to be performed in order to normalize it and this is the reason why we need to use the RMSLE as our evaluation metric.



**Algorithms and Techniques**

Given one of the purposes of this project is to explore advanced regression techniques, I am going to use the ensemble learning technique to achieve better evaluation score. The ensemble learning techniques aim to learn a weighted combination of the base models and it sometimes called a committee method (Murphy, 2012). The technique we implemented in this project is called stacking. Stacking, which stands for “stacked generalization”, was introduced by Wolpert (1992). The data science competition participants use this method extensively in order to win the competition. One well known example is the Netflix prize competition winning solution, BellKor’s Pragmatic Chaos.

The basic idea of stacking is to use another model or “stacker” to combine all previous model predictions in order to reduce the generalization error. An illustration of a 2 level 5 folds stacking approach is shown in Figure 6. First, the training dataset need to be split into 5 folds. Second, we iterate over this 5 folds training dataset. In each iteration, each base model will be trained using 4 folds and predict on the hold out fold. At the same time, each base model also need to provide a prediction on the entire test dataset. After the iteration over all folds, we will have the prediction of the entire training dataset for each model and 5 copies of the prediction of the entire test dataset for each model. Finally, we train second level model, or stacker, using the prediction in the training dataset as new features and use the average of the 5 copies of the test dataset predictions as the test input for the trained model to provide the final prediction.

|  |  |
| --- | --- |
| **Feature Engineering** |  |

At this stage we have to get variable by variable and impute the values if reasonable. One thing to have in mind is what does NaN stand for this specific feature and then impute it accordingly.

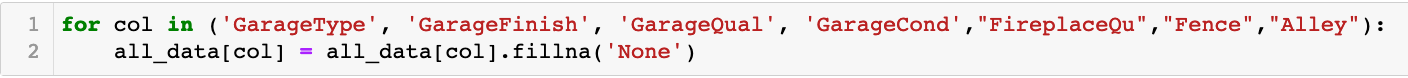
PoolQC: NA means "No Pool"



MiscFeature : NA means "no misc feature"



Now there are a couple of features where missing values indicates None (meaning it does not exist) so a for loop for these columns should do it:



*LotFrontage*: Linear feet of street connected to property. Now this property of the house is most likely going to be similar to the other ones in the neighborhood. So in this case they will be grouped and imputed with the median considering that might be some potential outliers.

Another thing that can be noticed is that some variables will inherit the imputed value due to the fact that we do not have the object at hand. For example having no garage implies for the variables ‘GarageYrBlt’, ‘GarageArea’ and ‘GarageCars’ have the NaN’s values meaning zero.

Same approach applies to the following variables: ‘BmstQual’, ‘BmstCond’, ‘BmstExposure’, ‘BmstFintType1’, ‘BmstFintType2’, which is fill them with zeroes.

Likewise for categorical values of basement NaN will imply "None"



Going further down the list with missing features. This is some important part that should be applied to variables with similar procedure.

NA most likely means no masonry veneer for these houses. We can fill 0 for the area and None for the type.



MSZoning: since "RL" is the most common values, we are going to use mode to it.

Utilities : This categorical variable has almost all of the observations as "AllPub", except for one "NoSeWa" and 2 NA . No predictive power so will be removed.



Functional: Read the data description or documentation, there we can see that NA means typical



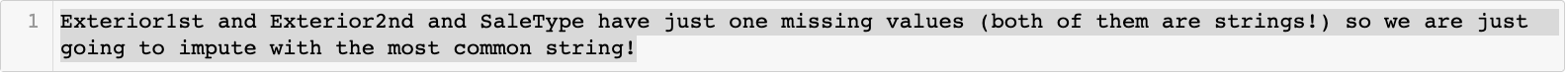
Moving down the list Electrical has one NA value. Since this feature has mostly 'SBrkr' (but not only), we can set that for the missing value.



KitchenQual: Same as for electrical, only one missing value can be seen and it will be imputed it with the mode.



Exterior1st and Exterior2nd and SaleType have just one missing value and both of them are strings so they will be replaced with the most frequent string.



The test example with ID 666 has GarageArea, GarageCars, and GarageType but none of the other fields, so they will be filled with the mode.

The test example 1116 only has GarageType but no other information. It will be assumed it does not have a garage.

A new variable will be created lot\_frontage\_by\_neighborhood which will be assigned with imputing missing values by the median LotFrontage grouped by “Neighborhood”.

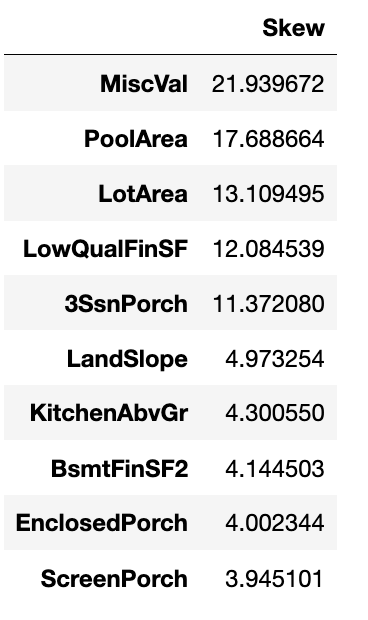
And finally after those modifications there are no missing values.

**Data Correlation**

Now the correlation matrix can be plotted and it shows between -1 and 1 weak and strong association between variables respectively and 0 meaning no correlation between variables, so after this step one crucial part is up next which is label encode and dummy encode considering as well fix skewness so it can be scaled and compared for modeling.

A new feature total square footage will be created at this point and as soon as we have numerical variables, check for skewness and correct it. There are 59 skewed numerical features to Box Cox transform.

Skew in numerical features:



**Modeling**

Before we proceed we should note that we did not scale anything until now. Scaling should take place before PCA or kNN for example because different metrics will affect the results. Also when performing gradient descend not scaling might slow-down the speed of algorithm. In Lasso and Ridge regression penalize the outliers, hence we need to make sure outliers are on the same scale (across all of the features).

**Base models scores**

Lasso score: 0.1115 (0.0074)

ElasticNet score: 0.1116 (0.0074)

Kernel Ridge score: 0.1153 (0.0075)

Gradient Boosting score: 0.1167 (0.0083)

Xgboost score: 0.1158 (0.0064)

We begin with this simple approach of averaging base models. We build a new class to extend scikit-learn with our model and also to leverage encapsulation and code reuse (inheritance). A way to look at classes is that they are now same type as our model. Meaning we define which functionalities they will have. Very simply said, averaging base method uses predictions (y\_prediction) average values of all the algorithms as our final prediction. In more complicated ensembling (further below) added a meta-model which is trained on the predictions from the previous ones. (Please differentiate between boosting (ensamble method) where we "stack" build up on residuals of the former-weaker models.

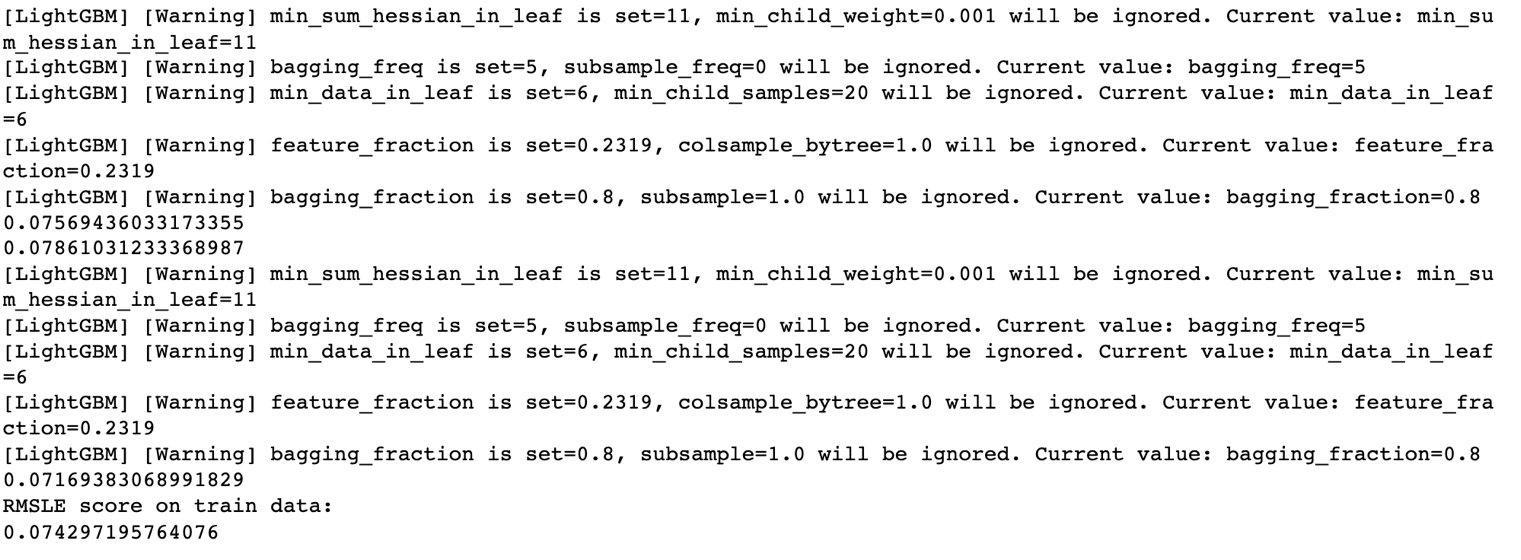
**Stacking**

It is an ensambling method used to make predictions of other models to make the final prediction. In short what happens is that moving one step toward the true solution (here with averaging of bagging method) and now for the final model (here it is xgboost) will be using actually predictions as predictor variables.

So now let’s say we have had on the initial M models n predictors that are used to make predictions. Than on the second model we will have M predictors (for M models) and the same number of rows in other words observations. But here is the crucial part, what observations?

K-Fold Cross Validation has to be used when training the first set of models. In order to make sure that we really exploit weaknesses and strengths of different models we need to know were are they strong or weak, if we were not to use k-fold cross validation than we would not find out.

Predictions from a validation (hold-out) set are going to be the new features of the final model



**Conclusion**

In summary, the ensemble approach in this project greatly improve the performance of the model in terms of the Kaggle Leaderboard score and in general approach compared individually which can be sure improved using some other advanced techniques. It shows some very accurate predictions for the business partners in the real state industry favoring both buyers and sellers.