Data Analysis Project

Arthur Krieff & Marc Vigneron

22/12/2019

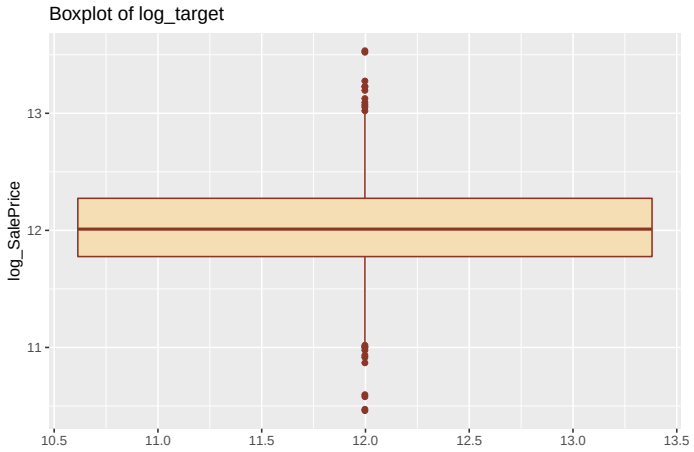
Introduction

This project is about the implementation on a real data set of the regression methods we reviewed in the course MAP535. The data set is taken from Kaggle competition House Prices: Advanced Regression Techniques. The goal of the project is to predict the price of houses from the testing set and to find out which are the most influent variables on the Sale price.

Exploratory Data Analysis / Initial Modelling

The data we are provided with has 1095 observations and 68 variables. Of those 68, 39 are categorical variables while 28 are numerical (+ the target).

With the help of an histogram, we observe that our target is right-skewed. Therefore, we log-transform it.

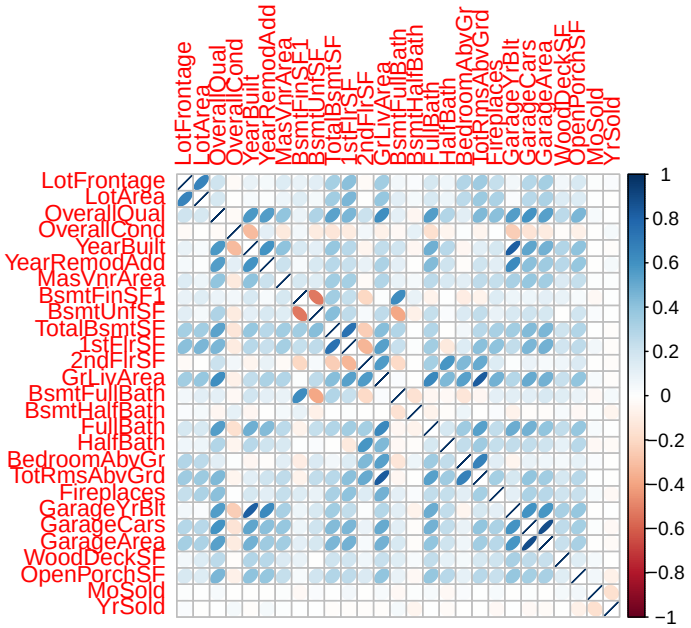


The boxplot of the transformed target shows many outliers. We will deal with these observations later on.

I. Numerical variables

First, let's focus on the 27 numerical variables (we do not consider the 'Id' variable). We notice that all provided observations are standardized and scaled. We use boxplots, histograms and other graphical and numerical tools to visualize the data.

We observe that:
- Some of those are heavily skewed ('BsmtHalfBath' and 'YearBuilt') ;
- Few seem Gaussian: only 'GrLivArea' and '1stFlrSF' look Gaussian.



Then, we examine the correlations of the numerical variables with a correlation-plot. We find that many pairs seem highly correlated, for instance:
- 'GrLivArea' and 'TotRmsAbvGrd' (correlation of 0.83)
- 'GarageYrBlt' and 'YearBuilt' (correlation of 0.82)
- 'GarageCars' and 'GarageArea' (correlation of 0.89)

We are also interested in how each covariate is correlated to our target.

	Correlation_SalePrice
OverallQual	0.8085330
GrLivArea	0.7303604
GarageCars	0.6776512
GarageArea	0.6353858
TotalBsmtSF	0.6002995
1stFlrSF	0.5950849
FullBath	0.5942059
YearBuilt	0.5925862
GarageYrBlt	0.5701173
YearRemodAdd	0.5658001
TotRmsAbvGrd	0.5509546

When looking at the correlation of log_target with the numerical variables, we note that approximately 11 variables are highly correlated (correlation higher than 0.5). Recall that many of these 11 variables are highly correlate with one another (e.g. 'GarageCars' with 'GarageArea' and 'OverallQual' with 'GrLivArea').

II. Categorical variables

We now move onto the 39 categorical variables.

We first observe that the plan is unbalanced. For several factors, note that some modalities are present a very low number of times, such as 'Utilities', 'Street', 'RoofMatl' , 'Heating' and 'Condition2'. Furthermore, according to the boxplots, the SalePrice value for these rare modalities is not very different from the mean. Therefore, we decide to delete these factors.

Then considering each categorical variable against log_target, we note with boxplots that some variables seem to have a great impact on log_target: for example 'MSSubClass', 'KitchenQual', 'CentralAir' and 'Neighborhood'.

Now, let's try to check the interactions between our factors and some numerical variables. We focus on the two numerical variables the most correlated with the log_SalePrice: GrLivArea and OverallQual.

After analyzing the linear regression plots based on the different factors and the two covariates (i.e. OverallQual and GrLivArea), we started building our intuitions as to their interactions with one another. For instance, it seems that, depending on the Neighboorhood, the GrLivArea has a higher or lower impact on the (log) SalePrice. This kind of interaction (difference in the slope coefficient) was also observered for the following couples of variables (non-exhaustive list):

Interaction with OverallQual: Neighborhood, BsmtCond, CentralAir and Functional.
Interaction with GrLivArea: MSSubClass, LandContour, ExterCond and BsmtFinType1.

Modelling and Diagnostics

I. Linear models

1. Without penalization

We start by building a multivariate linear regression model, taking into account all of the variables. However, we do not consider yet the interactions between the factors and the numerical variables. By default, the model creates dummy variables for our factors.

This model shows an Adjusted R-squared of 0.9163. Overall, there are 214 coefficients. Many variables have a very low p-value such as : Intercept, LotArea, OverallQual, OverallCond and GrLivArea for instance. However, we cannot trust this p-value as it does not take into account the correlations among the variables. For example, TotRmsAbvGr is highly correlated with the log SalePrice but the p-value is very high. This is because TotRmsAbvGr is also highly correlated with GrLivArea.

We can also notice that the p-value of the global Fisher Test is extremely low, so the test indicates that this model contains at least one variable that is relevant to explain the target.

The next logical step would be to select a smaller model where all (or almost all) variables are relevant since we have a lot of variables which the t-test seems to disqualify.

2. AIC/BIC criteria

To do so, let us perform a step-wise search with the AIC and BIC criteria.

AIC Criterion: the model selected is now much smaller than the previous one. The Adjusted R-squared is even higher: 0.9177. With the intercept, this model uses 138 coefficients. We can notice that highly correlated variables have now disappeared: for instance, out of the highly correlated pair TotRmsAbvGr and GrLivArea, only GrLivArea remains.

BIC Criterion: with this criterion, the model selected much less variables (25 coefficients). The Adjusted R-squared is a little bit smaller (0.8883) and all of the variables appear to be very significant according to the t-test.

Since it is hard to select one of the two models, let's try them on the testing set and see the performance.

	AIC	BIC
R2 Score (testing set)	-4.116113	0.8961199

While the BIC model shows a very good R2 score, the AIC model is clearly overfitting. Therfore, let's build new models that add a penalization term.

3. LASSO

First, we try the LASSO penalization on all our variables, still without taking into account the interaction terms. The model also selects the best lambda parameter by applying a cross-validation method. The model keeps 92 coefficients

and shows an R2 score of 0.885 and an RMSE of 0.135 on the training set.

	Beta
(Intercept)	11.8516104
NeighborhoodStoneBr	0.1553923
GrLivArea	0.1160398
NeighborhoodNridgHt	0.1111045
NeighborhoodCrawfor	0.0963492
NeighborhoodNoRidge	0.0907410
OverallQual	0.0776898
NeighborhoodSomerst	0.0736139
CentralAirY	0.0488675
FunctionalTyp	0.0465284

Since our data is standardized, we can easily interpret the importance of a variable based on its coefficient's value. Here, Neighboorhood, GrLivArea and OverallQual seem to be the ones that best explain the log SalePrice.

Applied on the testing set, the R2 Score is about 0.915. The best parameter for λ is 0.004.

4. Ridge

By applying the ridge model, the results on the training set are the following:

lambda	RMSE	Rsquared
0.1	0.1399768	0.8760609

The model considered 216 non-zero coefficients. Here, the most significant variables are 'Neighborhood', "Condition1" and "SaleTypeCon".

Applied on the testing set, the R2 Score is about 0.906.

5. Elastic Net

The Elastic Net model returns an R2 Score of 0.885 and an RMSE of 0.135. These results are associated with $\lambda = 0.02$ and $\alpha = 0.2$. The model ended up keeping 99 coefficients.

Here, the most significant variables are "Neighborhood", "GrLivArea" and "OverallQual", which is consistent with our Exploratory Data Analysis.

Applied on the testing set, the R2 is about 0.915.

6. PLS (Partial Least Squares)

By using the PLS model, we now have 14 components, which gives an R2 on the training set of 0.88.

So far, this is a summary of the performance of our models:

Methods.on.Testing.set	R2.Score	RMSE
Lasso	0.917	0.118
Ridge	0.906	0.125
Elastic Net	0.915	0.119
PLS	0.919	0.116
BIC	0.896	0.131

Overall, the PLS and Lasso models are the ones which perform the best on the testing set. Therefore, we will keep them and discard the others when pursuing our analyses.

7. Adding interaction terms

Now, let's try to complexify our linear models by adding the interaction terms. As seen in the Exploraty Data Analysis, the following interaction terms are worth considering:
- OverallQual-Neighborhood
- OverallQual-BsmtCond
- OverallQual-CentralAir
- OverallQual-Functional
- GrLivArea-MSSubClass
- GrLivArea-LandContour
- GrLivArea-ExterCond
- GrLivArea-BsmtFinType1

To analyse the interaction effects let's build anova tests.

According to the anova tests, all these interaction terms are significant except for OverallQual-CentralAir and OverallQual-BsmtCond. Let's try to add them to our models and check the performance.

LASSO

On the training set, the model gives an R2 around 0.89 and an RMSE of 0.132.

The R2 score on the testing set is 0.920, which is the best R2 Score so far.

PLS

On the training set, the model gives an R2 of 0.887.

Applied on the testing set, the R2 score is about 0.921.

Overall, these interaction terms seem to be relevant to our models.

II. Non-linear models

1. RandomForest Regression

For our first test of a non-linear model, we choose to perform a RandomForest Regression.

First, we ran the model with the default parameters given by R and then improved the model by performing several grid searches that we do not display here for brevity's sake.

Thus the parameters of our optimal RandomForest-Model are the following:

mtry	maxnode	ntree
113	68	300

And the scores of the optimal model on the testing set:

R.2	RMSE
0.8784565	0.1420062

We now move onto our next nonlinear regressor.

2. Gradient Boosting

We use here the Global boosting Machine Regressor from the gbm library.

Whereas random forests build an ensemble of deep independent trees, GBMs build an ensemble of shallow and weak successive trees with each tree learning and improving on the previous. When combined, these many weak successive trees produce a powerful "committee". Similarly, we first train the model with the default values and then improve it through successive grid searches.

We are going to search across 81 models with varying learning rates and tree depth. We vary the minimum number of observations allowed in the trees terminal nodes (n.minobsinnode) and introduce stochastic gradient descent by allowing bag.fraction < 1.

Multiple rounds of grid search indicated that the last best model we found had the following parameter:

ntrees	interaction.depth	shrinkage	n.minobsinnode	bag.fraction
2078	7	0.01	5	0.8

R2-Score	RMSE
0.9186145	0.1162024

We see here a significant improvement from the Random Forest Model since we obtain a R2 score of 0.918 and an RMSE of 0.116 on the testing set. We will now look at our final non-linear regressor: Extreme Gradient Boosting Regressor.

3. XGBoost Regression

XGBoost is an optimized distributed gradient boosting method designed to be highly efficient, flexible and portable. It implements algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. Since XGBoost only works with matrices containing only numerical variables, we will need to hot-encode our data. For this, we use the vtreat package.

We create our hyperparameter search grid along with columns to dump our results in. Here, we create a pretty large search grid consisting of 16 different hyperparameter combinations to model.

We train the result of our search on our train dataset. We obtain the parameters

max_depth	min_child_weight	eta	nrounds
7	2.3	0.01	5000

Our scores with the XGBoost Regressor on the testing set are:

R2.Score	RMSE
0.9073362	0.1239928

On the testing set, the XGBoost model returns an R^2 of 0.907. We need to be aware that hyperparametertuning for boosting method is very cumbersome, and time-consuming, which is why we did not push it until we'd receive an optimal model. Had we done that, it is very likely that our optimal XGBoost Model would have outperformed all others.

Final Models

Let's summarize our results:

Methods	R2	RMSE
Lasso	0.9202258	0.1150463
PLS	0.9213539	0.1142300
RandomForest	0.8784565	0.1420062
Gradient Boosting	0.9186145	0.1162024
XGBoost	0.9073362	0.1239928

The three best models are Lasso (with interactions), PLS (with interactions) and Gradient Boosting. We will continue our analyses only with these models.

I. Outliers

Since the Gradient Boosting algorithm is not very sensitive to outliers we will not study the presence of outliers for this model. So the final score for this model is: $R^2 = 0.907$.

As to the Lasso and PLS models, even though the BIC model is not the best performing one, we will check the presence of outliers based on the BIC model since there is no command in R to do a proper check of outliers for the LASSO and PLS models.

- **Cook's distance**: no observation has a value higher than 1, so according to this criterion we shouldn't remove any observation
- **Studentized residuals**: many values are regression outliers, especially observation 199 and 596.
- **Hat-values**: there is no observation higher than 0.5, so no observation has a large influence on its own estimation.
- By conducting an **outlier test**, we notice that observations 199, 596, 336, 633, 618, 323 and 692 are outliers and therefore should be removed.

Even though, an outlier on one regression model is not necessarily an outlier on another regression model, we could try to remove these observations in our Lasso and PLS models, and then check the new performance.

LASSO adjusted

This updated model returns an R2 on the training set of 0.923 and an RMSE of 0.1095 for $\lambda = 0.002$. By looking at the coefficients, it seems that the interaction term MSSubClass40:GrLivArea is quite influent on the SalePrice, as well as Neighborhood.

On the testing set, the R2 score is about 0.925.

PLS adjusted

The lasso model maximizes the R2 score on the training set when n_component = 9. It gives an R2 score of 0.921.

On the testing set, the R2 score is about 0.924.

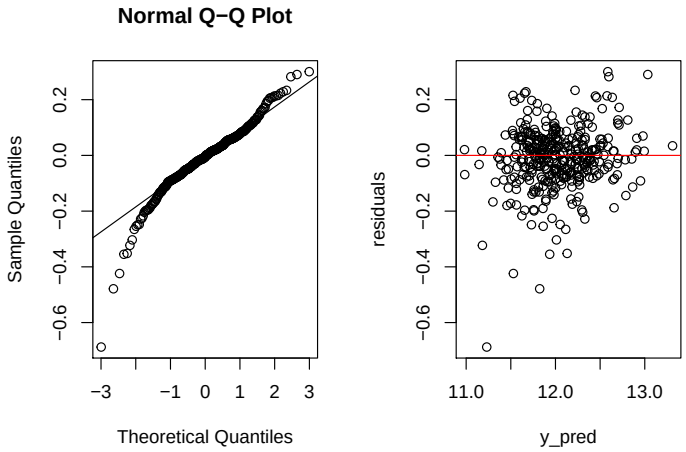
Summary

Methods	R2.Score	RMSE
Lasso	0.9247759	0.1117172
PLS	0.9253102	0.1113198

Based on these results, we have decided to keep the lasso model.

II. Model validation

Let's check if the assumptions of the lasso model are met.



- The residuals almost fall on the line (except for the tails), therefore they seem to follow a normal distribution.
- The residuals seem well spread around 0.
- According to a manually built Breush Pagan test, the variance of the residuals is homoscedastic (at the risk level $\alpha = 0.05$)
- As to autocorrelation, the value of the Durbin Watson test is around 2, which means that there is no correlation between the residuals.

Our lasso model is therefore **valid**. Let's recap the values of the RMSE and R2, this time by using the real values of the SalePrice (not the log_SalePrice).

Final.model	RMSE	R2
Lasso	21159.64	0.9324082

Discussion

Overall, we can point out several ways this analysis could be taken further.

First, it would be more appropriate to perform the outlier detection directly on the Lasso-model instead of using the BIC-model.

Also, we could not tune the XGBoost regressor as much as we would have liked, because of the huge amount of time it takes to do it. Therefore, a more advanced tuning of the hyperparameterswe would surely give a much better performing XGBoost model, which could maybe outperform the Lasso model.

Another interesting direction for further research would be to perform different kernelized regression models such as the KernelRidge regression.

Finally, a more advanced study of the effects of the interactions between the variables and the SalePrice would probably improve the model as well.