CS598 - Project 1: Predict the Housing Prices in Ames

Author: Gunther Correia Bacellar, NetID: gunther 6 Date: 10/10/2020

Introduction

In this project I analyzed the performance of different statistical models to predict the final price of residential properties (in log scale) in Ames, Iowa from 2006 to 2010 using the Ames_data.csv dataset.

Models analyzed were Linear Regression, Random Forest, GAM, Lasso Regression, Ridge Regression, combination of Lasso (for feature selection) and Ridge Regression, Elasticnet and XGBoost. The two models with best RMSE were analyzed in more details.

Pre-Processing Procedures and Analysis of Linear Models

Step 1: The dataset analyzed did not include many missing data, except for the predictor Garage_Yr_Blt, that showed NA values for houses without a garage. I replaced all NA for this predictor to zeros. Also, there is a value of Garage_Yr_Blt equal to 2207, that was probably a mistype error, so I replaced this value to 2007.

Step 2: There is a large number of 46 categorical variables (23 nominal, 23 ordinal) and I transformed them by generating K binary categorical variables when K>2 and K-1 for K=2 (K as the level for each categorical variable).

Step 3: I started my analysis taking the Linear regression as baseline because it is a quick and easy model to implement, but average test RMSE of all 10 splits was very high: 0.4093. Due to the presence of many variables in the dataset, I moved to Lasso regression, obtaining an average test RMSE of 0.1454, but still very high.

Step 4: The presence of extreme values in features with high correlation to Sale_Price is impacting the performance of RMSE, I continued the feature engineering process and applied a winsorization in 16 numerical variables to limit the effect of these extremes values. I computed the upper 95% quantile of these 16 training variables and replaced all values in training dataset (and later in the test dataset) bigger than these training quantiles. The effect of winsorization in the Lasso regression was a significative, reducing average test RMSE from 0.1454 to 0.1251.

Step 5: I removed 11 variables that I considered noise features and were impacting the RMSE. These features have some issues such as: no much variation in the values for all responses, variables with too many levels, variables with repetition values from other variables, or variables that I manually tested and didn't contribute to improve RMSE. The Lasso regression were not able to filter many of them effectively due to the elevated number of noise features in the dataset. This removal effect reduced the Lasso RMSE from 0.1251 to 0.1237, an improvement over the model applying only the winsorization (see figure 1). A similar behavior was observed for linear regression, with a drop in the RMSE from 0.4093 to 0.1360 after applying winsorization and removal of variables.

All feature engineering of test data used only information about quantiles and levels from training dataset, not the training dataset itself, and were applied after training the model using training data only.

Second Best Model: ElasticNet

Step 6: After incorporating winsorization and variable removal in the feature engineering process, I continued analyzing other linear models looking for improvement. As expected, Ridge regression had a higher RMSE (0.1248) than the Lasso Regression RMSE (0.1237).

Step 7: I tested the combination of Lasso regression for feature selection with ridge regression, using their built-in CV procedures to select tunning parameters, what generated a RMSE of 0.1232, using lambda.min for both regressions, slightly better than the Lasso RMSE.

Step 8: Finally, I tested the ElasticNet model with a range of values for alpha from 0.1 to 0.9, using lambda.min chosen by CV as well as all feature engineering applied before. The best result was for alpha = 0.2 (figure 2), with a CV RMSE of 0.1224, and all splits with RMSE below 0.125 except splits 6,7 and 9, which still had RMSE below 0.135.

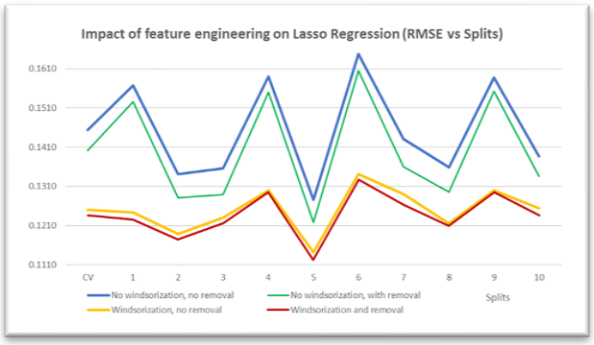

Figure 1: Impact of winsorization and removal of variables in the RMSE

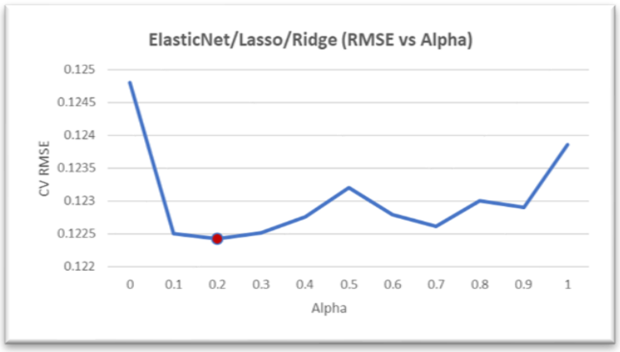

Figure 2: CV RMSE for different values of alpha. Alpha=0.2 had the best result

For Lasso, Ridge and ElasticNet regressions analyzed in this project I used their built-in CV procedures to select tunning parameters. Initially I tried lambda.1se chosen by CV, but it returned a RMSE on average 7% larger than the one using lambda.min, so I opted use lambda.min in all my analysis.

Best Model: XGBoost

Tunning XGBoost model was more complex and time consuming than tunning the different linear models due to the high number of parameters. For this dataset, I prioritized 5 XGBoost parameters for tunning (Vasconcelos, 2018):

- **eta** (shrinkage): how much information from a new tree will be used in the Boosting
- **colsample_bylevel:** re-sample the variables in each new node
- **max_depth:** control the max depth of the trees
- **subsample:** define if we are estimating a Boosting or a Stochastic Boosting
- **gamma:** manage the minimum reduction in the loss function required to grow a new node in a tree.

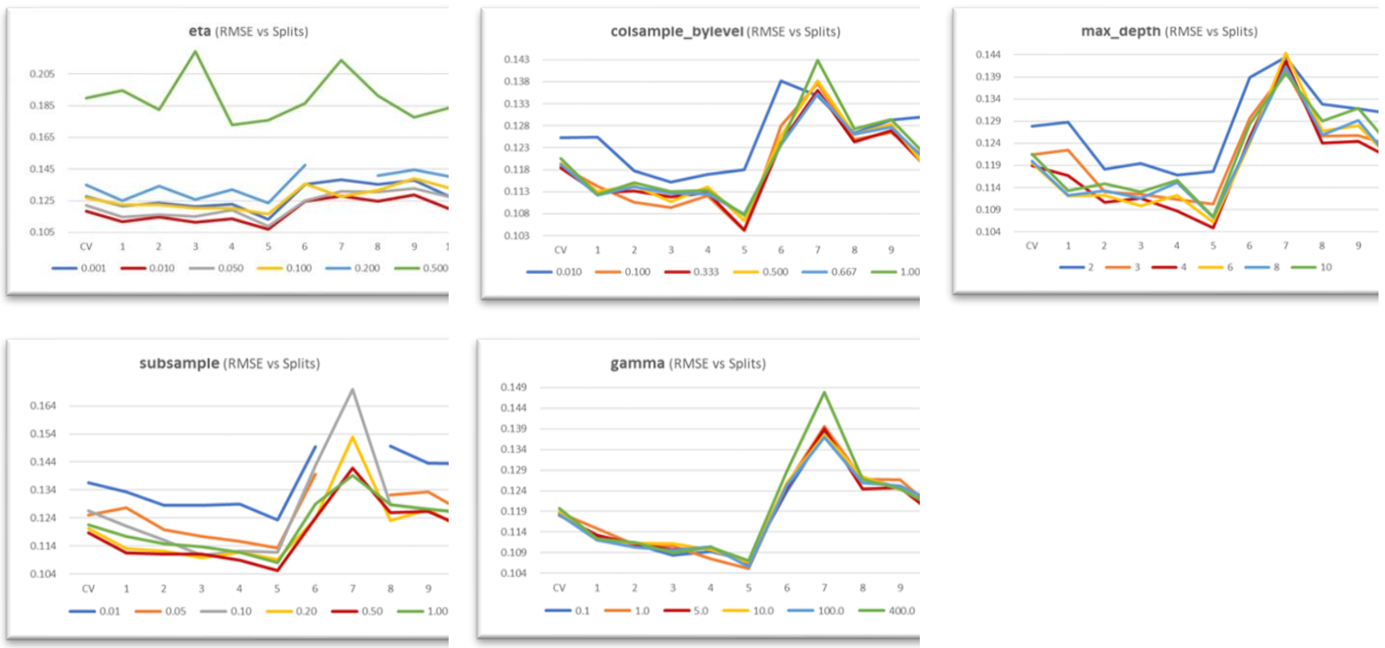

Figure 3: Cross-validation to select the lowest RMSE for 5 different parameters. In Red the graph with best CV RMSE.

Step 9: I started with standard values: colsample_bylevel = 2/3, max_depth = 6, subsample = 0.5, and gamma=1. Then, I used CV for each range of parameters (6 values per parameter). As soon I found the best CV RMSE and identified the value of the first tunning parameter (eta), I adopted it, and moved to a new range of values for the second parameter (colsample_bylevel), repeating the process until I had tuned all 5 parameters for all 10 training/test splits. In this tunning process I fitted a total of 300 different XGBoost models, selecting the value of each parameter using cross-validation, requiring more than 1h30 of computer processing time. See figure 3 with the curve of all different parameters and values tested for each training/test split. Using cross-validation, the final tunning parameters found were: eta = 0.01, colsample_bylevel = 0.333, max_depth = 4, subsample = 0.50 and gamma = 5.0, configuration that produced a test RMSE of 0.1176. As standard value, I adopted nrounds = 4000.

Step 10: I also run XGBoost with winsorization and variable removal, but the test RMSE of 0.1200 was higher than when I didn't apply them (0.1176), showing that these feature engineering in steps 4 and 5 were not necessary.

Accuracy table (RMSE log Sale_Price)

Model	Time (*)	Split 1	Split 2	Split 3	Split 4	Split 5	Split 6	Split 7	Split 8	Split 9	Split 10
XGBoost	2.52 min	0.1140	0.1120	0.1121	0.1095	0.1061	0.1276	0.1263	0.1244	0.1232	0.1205
ElasticNet	20.98 s	0.1225	0.1177	0.1203	0.1197	0.1114	0.1333	0.1258	0.1195	0.1301	0.1240

(*) Computer processing time for training and testing 10 splits using a Surface Book 3, Intel Quad-Core 10th Gen i7-1065G7 CPU @ 1.50GHz, 32GB, with Windows 10 64-bit

Conclusion

Figure 4 shows a summary of all models tested in this project. Only Lasso, Ridge, the combination of Lasso and Ridge, Elasticnet and XGBoost returned a test RMSE below 0.125 for the first 5 splits and below 0.135 for the last 5 splits.

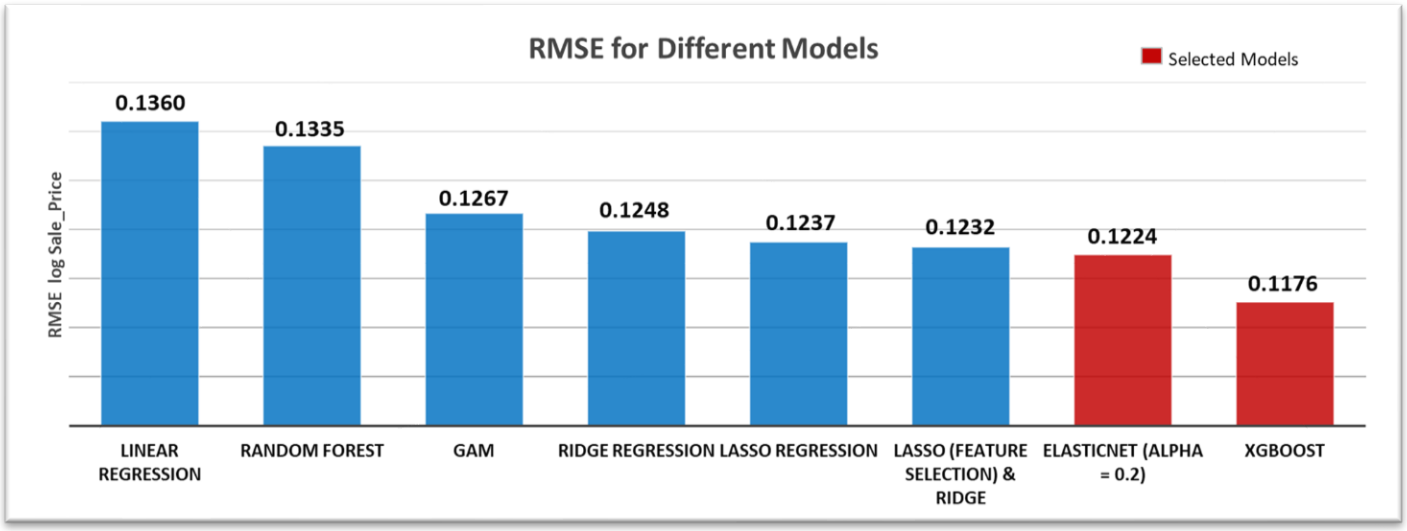

Figure 4 : Summary of CV RMSE for Different Models analyzed in this project

XGBoost model had the best performance and did not required feature engineering, but I spent more time, specially processing time, in the tunning process. The time processing Random Forest and tunning the parameters was even higher, but with a test RMSE performance worse than all linear models except the linear regression. The GAM process was also very time consuming, and the tentative to find a good combination of variables, transformations and parameters for the GAM model showed to be complex for a dataset with such large number of variables. Also, winsorization and variable removal were an important to improve the performance of all linear models.

References

Vasconcelos, Gabriel "Tuning xgboost in R: parts I and II"

https://insightr.wordpress.com/2018/05/17/tuning-xgboost-in-r-part-i/

https://insightr.wordpress.com/2018/07/28/tuning-xgboost-in-r-part-ii/

De Cock, D. (2011). "Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project," Journal of Statistics Education, Volume 19, Number 3.