Colorado State University

“House Prices - Advanced Regression Techniques” Kaggle Write Up

Ty Hammond, Blake Hammarstrom

DSCI478: Capstone Group Project in Data Science

Dr. Emily King

February 19, 2025

The Kaggle competition “House Prices - Advanced Regression Techniques” provides a dataset consisting of variables that describe different aspects of homes. The goal of this competition is to use the dataset provided to predict the final price of each house and this is how the submission is judged. We attempted to utilize multiple different regression techniques such as Ridge, LASSO, XGBoost, and Random Forest. The technique that resulted in the best result or most accurate predictions was Random Forest. This resulted in a Log Root-Mean-Squared-Error of 0.06829742. Throughout the rest of this write-up, the problem and competition will be described in depth, our process through this competition will be shown along with the rationale behind it, and our solutions and conclusions will be presented.

The Kaggle competition gives us a dataset with 79 explanatory variables describing aspects of homes in Ames, Iowa. Using this dataset we must employ regression techniques to predict the final price of each house. The submission will be evaluated on Log Root-Mean-Squared-Error or RMSE which is the difference between the logarithmic of the predicted value and the logarithmic of the observed sales price (Anna Montoya and DataCanary). Also on this Kaggle page are tutorials covering EDA, techniques such as random forest, XGBoost, LASSO, neural net using the caret package, and more. This helped provide a guide on possible techniques we could employ and gave us a foundation to build on.

The process behind the models began with importing the training and test data set and performing exploratory data analysis. We did an overview of the structure, and summary statistics, and searched for missing values in the data.

We separated the target variable “SalePrice” from the training data set to ensure whatever models used were trained only from the other predictors. The dataset had a mix of numerical and categorical variables so we converted the categorical variables into dummy variables using caret.

This was an important step because most machine learning models, especially random forest does not allow categorical input. After converting the data we then had to align it to ensure the training and test set had the same number of dummy variables. This was a continuous problem throughout the project but after extensive debugging time, we were able to get the data aligned. To adjust for the missing values in the data we used median imputation. We did this because firstly, we did not want to exclude data and lose power in our model, second, mean imputation is very sensitive to outliers while median imputation is not. The next step was model training. We initially started with LASSO and Ridge techniques to provide a baseline as they are simple and linear. We then moved into the more complex models which were Random Forest and XGBoost. For the Random Forest model, we used 100 trees. We then made our predictions on SalePrice using the test dataset. After this, we were able to get final predictions that matched the expected output and appeared reasonable. This gave us final predictions in the form of an associate id and predicted sale price which was the format that Kaggle wanted us to submit our results in.

	Id <dbl>	SalePrice <dbl>
1	1461	128426.4
2	1462	156158.0
3	1463	173426.6
4	1464	187580.2
5	1465	196010.3
6	1466	185975.6

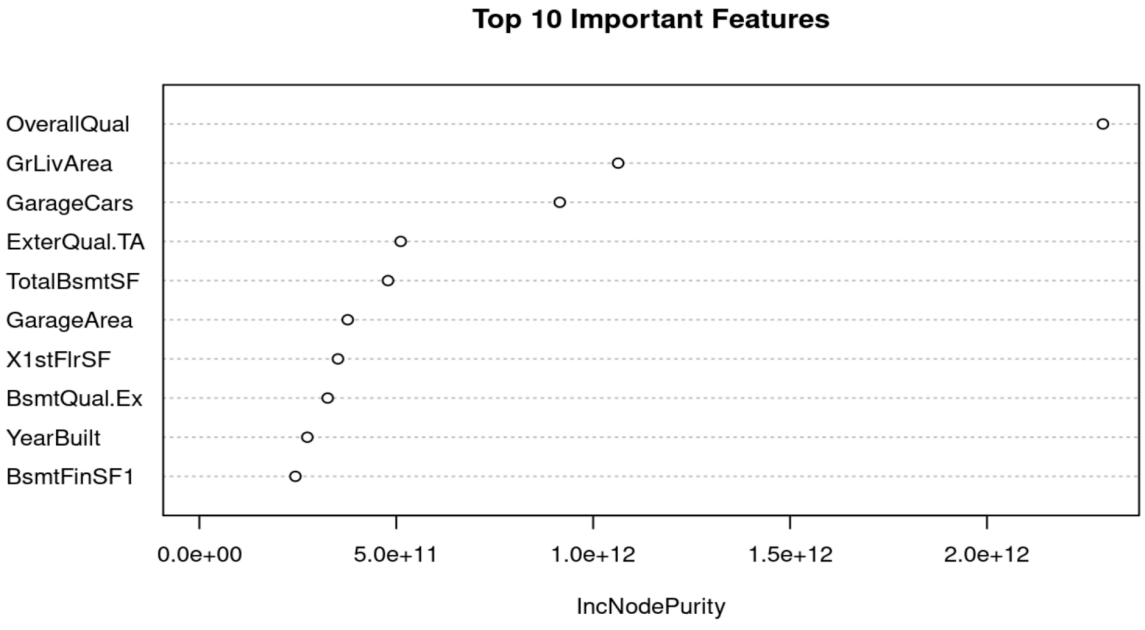
Without submitting to Kaggle we wanted to get our RMSE and log RMSE values. To do this we create an 80/20 training validation split. We then predicted on the validation set as well which allowed us to get our RMSE and log RMSE values.

To analyze model performance, we computed the RMSE and log(RMSE) on validation sets for each of our approaches: Lasso, Ridge, Random Forest, and XGBoost. Our analysis

revealed that the Random Forest and XGBoost models performed the best having log(RMSE): 0.0683 and 0.0854 respectively. This suggests that these models are better at capturing the relationships in our data. We also experimented with different seeds but found similar results and Random Forest was consistently the best performing model.

Random Forest was our best model and to put the conclusion into practical terms, our RMSE, not log RMSE was 13473.04 which means that on average, the predicted sale prices deviate from the actual sale prices by about $13,473.06. This is a very reasonable assessment for our model and with home prices varying greatly, the model was extremely accurate. The percent of variance explained by the model was 85.56% which means that 85.56% of the variance in the target variable is explained by the model.

The importance plot showed the top 10 most important features in the Random Forest model, with overall quality(Overall material and finish quality) being the most influential for determining the target variable. Other variables such as GrLivArea(Above Ground Living Area (sqfeet)) and GarageCar(Number Car Garage) also played a significant role in the target variable but after that, the importance scores became continuously less influential.



Our two linear models both performed relatively poorly compared to the more complex models. LASSO had a log RMSE of 0.1229 and Ridge had a log RMSE value of 0.1209. This was expected and we just used these simple linear models as a baseline to compare our more complex models.

While we had success with all of our models, to really get a foot in the competition we could explore other tuning techniques such as grid search, Bayesian optimization, or incorporating interaction terms to improve the model's power. Ultimately, we are happy with our results, as the models' performance, high variance, and low RMSE provide reliable predictions for our target variable.

Works Cited

Anna Montoya and DataCanary. House Prices - Advanced Regression Techniques.

Link: Kaggle Notebook

Kaggle Notebook on Mean and Median Imputation

Link: Kaggle Notebook

Mitsde Blog Post on Data Imputation Techniques

Link: Mitsde Blog Post

CRAN Documentation for impute_median Function

Link: CRAN impute_median Documentation

Caret Pre-processing Documentation

Link: Caret Pre-processing