Kaggle Final Report

Hannah Ritchie

Trinity University

Machine Learning

Table of Contents

[Introduction 3](#_Toc130402618)

[Kaggle Website 3](#_Toc130402619)

[Original Dataset 3](#_Toc130402620)

[Milestone 1 - Cleaning 3](#_Toc130402621)

[Handling Missing Values 3](#_Toc130402622)

[Categorical 3](#_Toc130402623)

[Numerical 4](#_Toc130402624)

[Correcting Typos and Measurement Errors 4](#_Toc130402625)

[Recoding and/or Augmenting Existing Variables 5](#_Toc130402626)

[Dealing with Outliers 5](#_Toc130402627)

[Transformations 5](#_Toc130402628)

[Creating New Columns 5](#_Toc130402629)

[Removing Columns 6](#_Toc130402630)

[Kaggle Prediction Note 6](#_Toc130402631)

[Milestone 2 6](#_Toc130402632)

[Plain Vanilla Model – Without Prescriptions 6](#_Toc130402633)

[Multicollinearity 6](#_Toc130402634)

[Nonconstancy in Variance Terms 7](#_Toc130402635)

[High Influential Points 7](#_Toc130402636)

[Milestone 3 and 4 – Modeling 7](#_Toc130402637)

[Base Model – After Prescriptions 7](#_Toc130402638)

[Forward AIC 8](#_Toc130402639)

[Backward AIC 9](#_Toc130402640)

[Hybrid AIC 9](#_Toc130402641)

[Ridge Regression 10](#_Toc130402642)

[Lasso Regression 10](#_Toc130402643)

[Simple Tree 10](#_Toc130402644)

[Bagging 10](#_Toc130402645)

[Random Forest 11](#_Toc130402646)

[Boosting 11](#_Toc130402647)

[Best Model 11](#_Toc130402648)

[Boosting 11](#_Toc130402649)

[Final Conclusion 13](#_Toc130402650)

# Introduction

## Kaggle Website

Kaggle is a website that sources many competitions involving a variety of data science topics. The competition that this paper will cover is Kaggle’s *House Pricing – Advanced Regression Techniques* competition. Based on the cleaning and regression techniques utilized, a user can input a prediction dataset to receive a score. The lower the score, the closer to the actual housing price score and vice-versa. The Kaggle website also has a leaderboard for scores to give a metric of how your scores relate to other users.

## Original Dataset

The original data set is 2 data sets. Through the Kaggle website you can download both a train and test dataset. For this project, I combined those data sets to more efficiently clean and to remove redundancy. In the newly combined data set, there is a total of 81 columns and 2,919 rows. Across all 81 columns, there is a total of 13,965 null values. Throughout the cleaning process, columns and a few rows will be dropped.

# Milestone 1 - Cleaning

## Handling Missing Values

### Categorical

The process for handling missing values is similar across almost all categorical columns, but there are two subcategories that it can be separated further into: dependent columns and independent columns. Dependent columns are columns in which values are dependent on having a certain attribute. For example, the **BsmtQual** column’s value are dependent on whether that house has a basement. Independent columns are columns in which values are not dependent on an attribute.

For the Dependent Columns, I filtered the null values and saw if an identifier column showed that it had no attribute or otherwise. If the null value was in a row that showed it did not have the attribute necessary to have a value in that column, I inputted “No” plus the attribute name. In the example of the Basement Exposure column, I filtered by null values and selected to show only the Exposure and Condition column.

For the Independent Columns, I used the summary function to find the most common value and inputted that value for the null. For this data set, there were no events of multiple values being equally common, so it was a very simple procedure.

### Numerical

The process for handling missing values is similar across almost all numerical columns, like categorical columns. There are two subcategories that it can be separated further into as well: dependent columns and independent columns. Just like categorical, dependent columns are columns in which values are dependent on having a certain attribute. Independent columns are columns in which values are not dependent on an attribute.

The process for inputted values for numeric dependent columns is incredibly like categorical dependent columns. Filtering the data was integral to inputted values that made sense in the scope of the column. The values inputted into the null values were the mean of the filtered data (removing nulls of course). In the event that the numeric column was in reference to an attribute that they did not have, I inputted the value 0. For independent numerical columns, the mean of that column (without nulls) was inputted.

## Correcting Typos and Measurement Errors

All typos and measurement errors were fixed in recoding and dealing with outliers. As such, I will devote my energy in explaining that process in their respective parts.

## Recoding and/or Augmenting Existing Variables

It was only after starting my initial modeling that I ran into issues with some of the column’s factors. Since some of the columns have factors that have an insufficient number of observations, I ran into errors in predictive modeling. That is, I had an influx of factors that had no observations in the practice set. For example,

## Dealing with Outliers

The removal of outliers was primarily done through boxplots to determine the break in the values (or an actual gap) in the boxplot. Each numerical column was tested for outliers before and after potential transformations to be thorough.

## Transformations

Like dealing with outliers, the need for transformations was determined by running a histogram graph for the numerical columns. If the column showed an approximately normal distribution, no transformations were run.

If the column was not normally distributed, depending on the skew, different transformations were tested to determine which led to a normally distributed histogram graph. Once the best transformation was found, the column was renamed based on the transformation (i.e., Log10ColumnName) to reduce redundancies.

## Creating New Columns

After the individual cleaning of each column is completed, numerical columns that could be logically grouped together were combined to reduce collinearity issues, redundancies, and to create more efficient predictor columns. The more specific criteria used to determine if a new column was necessary was if there were multiple columns that could be combined to create a full picture of the house. For example, there were four different Porch Square Foot variables – since they could logically be combined into a Total Square Porch column, it made sense to combine it.

There were four new columns created following that criterion: Total Home Square Footage, Total Porch Square Footage, Total Full Baths, and Total Half Baths. The variables used to create those new columns were removed and the skew was checked using a histogram graph.

## Removing Columns

The removal of categorical columns was done based on column makeup. That is, if a column was made primarily of one variable, the column was dropped. Numerical columns, again, were only dropped in the cleaning stage if it was used in the creation of new columns to reduce collinearity issues.

## Kaggle Prediction Note

It's important to note that since I transformed the SalesPrice Column in the data using a log transformation, I had to undo that transformation before turning in my predictions to the Kaggle website.

# Milestone 2

## Plain Vanilla Model – Without Prescriptions

Before any of the prescriptions outlined in Milestone 2, the Plain Vanilla Model predicted moderately well at a best score of 0.13857. The R-Squared of the model sits at a 0.9328 and the Adjusted R-Squared sits at a 0.9211. The R-Squared values having a difference of 0.0117 is an additional sign that the model does a moderately well job in prediction in the training data set. The F-Statistic is sitting at 79.88 on 216 and 1243 Degrees of Freedom and its associated P-Value at less than 2.2e-16 means that we can reject the null hypothesis that there is no relationship between predictors and *Log10SalePrice*.

## Multicollinearity

The issue of multicollinearity was a tough problem to prescript. Two functions were used to find columns that contained the issue of multicollinearity: alias and VIF. Alias was used to find issues of collinearity between categorical columns to allow the VIF function to run. After the VIF function could run with no error, it was a matter of finding columns with a high (>2) VIF value and looking at the adjusted r-squared value in a new model without that column. If the new model’s adjusted r-squared is greatly impacted (>.1) then the column is left in the model, because it greatly impacts the predictions scoring.

## Nonconstancy in Variance Terms

Since some of the prediction and the predicted columns were transformed previously for skew issues, I ultimately decided not to prescript the model further - the models used were within a respectable range to not prescript further.

## High Influential Points

Like multicollinearity, influential points were hard to prescript. I used the cooks distance plot to see if there were any highly influential points, but since there were points with leverage equal to one, the plot didn’t show any relevant information. I did some research online and ultimately decided to remove the points that lead to the leverage being equal to one. After that, I reran the model to rerun the cooks distance plot – since I ran into some additional leverage one points, I repeated that process. After I was able to run the cooks’ distance without errors, I was able to see that the model had no highly influential points.

# Milestone 3 and 4 – Modeling

## Base Model – After Prescriptions

After the issues of Collinearity and Highly Influential points were fixed, the new model had a best score of 0.13834. The best score didn’t change much from plain vanilla model we ran before prescriptions, but the distance between the R-Squared value and Adjusted R-Squared value decreased to 0.0105. Additionally, there were no longer any predictor values that had null values for the Estimate, Standard Error, T-Value, and P-Values. This shows that through the prescriptions, there is an increase in model accuracy and fit. This is additionally reinforced by the F-Statistic of 88.64 on 191 and 1263 Degrees of Freedom and its associated P-Value of less than 2.2e-16. This jump in the F-Statistic makes me more confident in the rejection of the null hypothesis.

## Forward AIC

This model did moderately well in predicting the SalesPrice with a best score of 0.13486 leading to a 0.0348 difference to the next best model (the base model). Thirty-two columns were determined to be significant through this method, so those 32 predictors were used in the linear model. The list of important predictors are as listed: Home Total SF, Overall Quality, Neighborhood, Garage Area, Year Remodel Added, BsmtFinType1, Above Ground Living Area, MS Zoning, Functional, Square Root Lot Area, Year Built, Sale Condition, Heating QC, Condition 2, Fireplace Quality, Basement Quality Garage Type, Condition1, Exterior1st, Kitchen Quality, Total Full Bath, Building Type, Total Half Bath, Lot Configuration, Log10TotalPorchSF, Wood Deck SF, Foundation, Sale Type, Paved Drive, Masonry Veneer Type, Roof Materia, Lot Frontage

The R-Squared value associated with this model is 0.9278 and Adjusted R-Squared value is .92. The distance between the R-Squared and the Adjusted R-Squared decreased to 0.0078 with this new model. The F-Statistic is 118.8 on 142 and 1312 Degrees of Freedom with an associated P-Value of less than 2.2e-16. Those two statistics in tandem explain why the score is getting better, in that, the model is becoming less overfit between the training and test datasets and less important predictors are being removed from the model.

## Backward AIC

This modeled performed slightly worse than the Forward AIC model with a best score of 0.13541. This can be easily explained through the statistics we mention for each model. The R-Squared and Adjusted R-Squared have values of 0.9264 and 0.9194 respectfully leading to a difference of 0.07. The difference is slightly decreased but leads to a less than optimal decrease in the Adjusted R-Squared value. This method garnered two less important predictors than the Forward AIC – the model has 30 predictors. The list of important predictors are as listed: Home Total SF, Overall Quality, Neighborhood , Garage Area, Year Remodel Add, BsmtFinType1, Above Ground Living Area , MS Zoning, Functional Square Root Lot Area, Sale Condition, Heating QC , Condition2 , Fireplace Quality, Basement Quality, Garage Type, Condition 1, Exterior 1st, Kitchen Quality, Total Full Bath, Building Type, Total Half Bath, Lot Configuration, Log10 Total Porch SF, Wood Deck SF, Foundation, Sale Type, Paved Drive, Masonry Veneer Type, Lot Frontage.

## Hybrid AIC

Hybrid AIC had the same best score as Backward AIC at 0.13541. This way of modelling also had the same coefficients as the Backward AIC method. The list of important predictors are as listed: Home Total SF, Overall Quality, Neighborhood , Garage Area, Year Remodel Add, BsmtFinType1, Above Ground Living Area , MS Zoning, Functional Square Root Lot Area, Sale Condition, Heating QC , Condition2 , Fireplace Quality, Basement Quality, Garage Type, Condition 1, Exterior 1st, Kitchen Quality, Total Full Bath, Building Type, Total Half Bath, Lot Configuration, Log10 Total Porch SF, Wood Deck SF, Foundation, Sale Type, Paved Drive, Masonry Veneer Type, Lot Frontage. Like the last method, this method garnered two less important predictors than the Forward AIC – the model has 30 predictors.

## Ridge Regression

Unlike the previous models, this model is much harder to interpret, although, it has a best score of 0.13431 – the best score so far. The best lambda associated with the model is 0.01718853. To make interpretation a little easier, I calculated the R-Squared Value. Although the calculated value of 0.922164 is lower than all other R-Squared it outperformed all other previous models. Only a few coefficients for variables reached zero: Roof Material - Membran, Roof Material - Metal, Roof Material - Roll, and Electrical - Mix.

## Lasso Regression

The best score of this model is great at 0.13145. Like Ridge Regression, Lasso Regression is difficult for me to interpret, however since a lot more coefficients reach zero. Since there a many more variables that reach zero, they will not be listed.

## Simple Tree

Simple Tree was the worst model at predicting with a best score of 0.2173. Since the Simple Tree Model had the same number of splits as the Pruning, there was no room for growth for this model. The Simple Tree grown is shown in the figure to the right.

## Bagging

The Bagging did adequately with a best score of 0.15352. The R-Squared value (or % variance explained) of this model is 86.28%. This variance compared to the previous linear models can explain the stark difference in best score. The most important variables according to the percentage increase in mean squared error if removed is Total Home Square Footage, Neighborhood, and Overall Quality.

## Random Forest

The Random Forest did much better than the Simple Tree and Bagging models with a best score of 0.14659 after finding most optimal *mtry* value. Although, compared to the Linear Models run, it is not even close to first. Before running the most optimal *mtry*, the R-Squared is at the value 87.4%. The most important variables according to the percentage increase in mean squared error if removed is Total Home Square Footage, Neighborhood, and then Overall Quality.

After running the most optimal *mtry* (which value is 14), the R-Squared sits at 87.79%. The most important variables according to the percentage increase in mean squared error if removed is still Total Home Square Footage and Neighborhood, but Above Ground Living Area is new.

## Boosting

Out of all the Tree-Based Methods, boosting has the best performance with a best score of 0.1299. The found interaction depth was 3 and the number of trees was 3,346. The variables with the highest relative influence are as listed: Total Home Square Footage, Neighborhood, and Overall Quality.

# Best Model

## Boosting

According to the Kaggle score I received for Boosting; my model is a slightly ideal model. Although it could get a better score with interaction terms, it does a pretty good job at predicting the Sales Price of a house. The Relative Influence, or the influence it has on the effectiveness of the model if removed for the first ten variables from the boosting model is in the figure below.

|  |  |
| --- | --- |
| Variable | Relative Influence |
| Home Total SF | 32.01269426 |
| Overall Quality | 23.32365399 |
| Neighborhood | 18.02301729 |
| Kitchen Quality | 2.61924362 |
| Garage Area | 2.26052535 |
| Gr Living Area | 1.87126670 |
| Exterior Quality | 1.64321075 |
| Year Remodel Added | 1.53626724 |
| Month Sold | 1.39928662 |
| Sqrt Lot Area | 1.17204998 |
| Year Built | 1.12320188 |

Since this model is not a linear model, it is impossible to interpret the coefficients of the variable and its actual numerical influence in Sales Price. Through the figure above, though, we can see that the most important variables that determine home price is Total Square Footage, Overall Quality, and Neighborhood. This conclusion is not a logical jump – that is it makes sense that those variables would have a much higher influence than any other factor. This means that throughout the training data (or historical dataset), we can determine that those three variables were the most important to the people buying the houses and influenced the Sales Price the most.

There were 37 predictors that had a Relative Influence of less than 1. The bottom three predictors, or the predictors that had the least impact on model efficiency were Condition 2, Masonry Veneer Type, and Roof Style. This indicates that those three predictors were the less important predictors for determining Sale Price. That conclusion is not surprising because those variables are mostly dependent on other variables.

Something that surprised me through the boosting method was the placement of Year Sold. It was relatively low on the Relative Importance Scale with a value of 0.19531469. With the historical knowledge of the housing bubble pop of 2008, I thought that that column would have a greater impact to the Sale Price.

There were a few predictors that were important/significant in the Linear models that turned out to be mostly unimportant in determining Sale Price. Those columns are as follows: MS Zoning, Building Type, and Total Porch SF. The same is said for vise-versa cases – following columns were deemed important in boosting (greater than 1 in relative influence) but were not significant in the Linear Models. Those columns were: Month Sold and Roof Material.

# Final Conclusion

Although Boosting is a little less straightforward to interpret, it results in the best predictor model because of the non-linear relationship between the predictors and Sale Price. According to the Kaggle Website, my best score of 0.12993 is around the First Quartile and my current leaderboard position is 932 out of 4250 – the top 22%.

I ran into quite a few problems in the later stages of modeling because of my inadequate cleaning at the start. If I was to go back in time and restart the project, I would spend significantly more time in making sure that my initial cleaning was more thorough. The main problems I ran into was not removing columns with a lot of entries in one factor and not combining factors that have lower number of entries. I fixed the factor problem before starting with the Tree-Based Methods by using the *fct\_collapse* function. The logic I used to combine factors was to leave factors with a large number of entries alone and combining the rest in pairs or threes making sure to have a logical combination of entry type and number. Once that issue was prescribed, my best score for my plain model went from 0.14 to 0.13.

I had an overall great experience on this project. Since I didn’t have the Kaggle project in my Business Analytics class, I experimented a lot with the Kaggle website. The project itself was a definite task – it took me most of the project timeline to complete milestone one and two, but it was an excellent challenge and learning experience. I feel much more confident in the topics taught during this semester and the topics taught in Business Analytics because of the project. Not only was the project incredibly fun, but it was also incredibly challenging. Because of the outcome of this project, I am incredibly excited for the next project.