Project 1: Predict the Housing Prices in Ames

Alfredo Castillo (arc13) , Seema Malviya (seemam2), Suhas Bhat (suhasb2)  
October 17, 2022

**Summary of Approach**

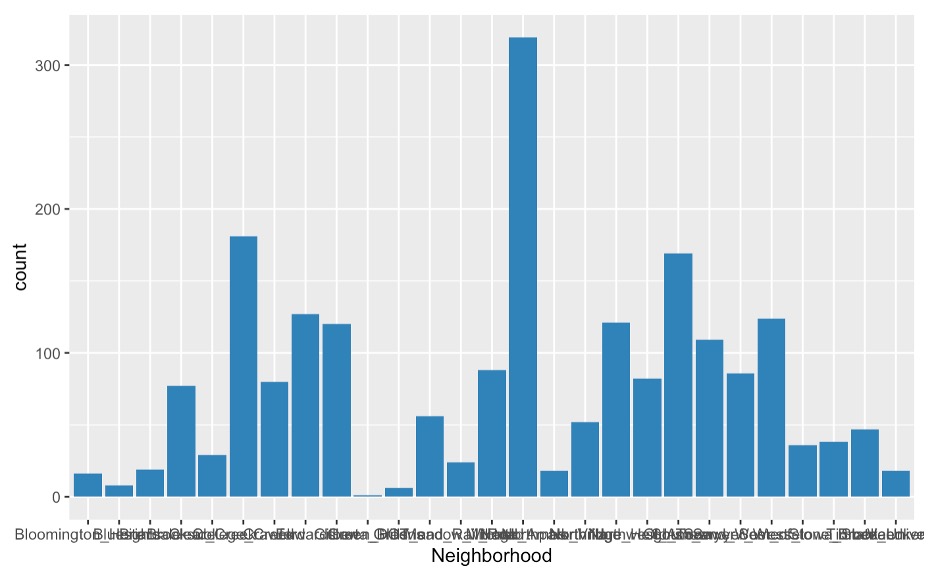
**Data Processing**

Prior to doing any model fitting using the House Prices in Ames dataset, it was required to performed a data cleansing process with the objective to identify incorrect, irrelevant, incomplete and other “dirty” parts of the dataset and then replace or cleanse the data.

The following steps were done for both the Training and the Test date sets.

* Data missing or dirty data: Inspection of the dataset showed several samples with NA. In particular, in one of the data sets splits, column 52 corresponding to "Garage\_Yr\_Blt" had 108 sample of NA values. Similar observation was made in other dataset splits. In order to clean these “dirty” data points, two approaches were tested: 1) replace "Garage\_Yr\_Blt" = 0, or 2) replace "Garage\_Yr\_Blt" with the value provided by the “Year\_Built”. The rationale was to assume that the garage was built the same year the house was built. Prediction accuracy results showed a significant impact with XGBoost method. Therefore the “Year\_Built” approach was selected.
* Irrelevant Observations: There were some variables in the dataset that were not relevant to the problem. In particular, the dataset had a column named PID. Which is important to identify a house when creating a report with the predicted sales price, but it not relevant to predict the sales price itself. Therefore, it was removed from the dataset used during the fitting and prediction process.
* Imbalance Categorical Variables: The next step was to remove categorical variables where one category may accommodate 95% or more of the total observations, which is not very informative and may cause an imbalance and overfitting.

As an example, a distribution of the “Utilities” variable, shows that more than 95% of all the observations were “AllPub” type. Whereas the “Neighborhood” variable had a distribution that provides more relevant information. See the following plots for comparison.



After making a few tests, the following recommendation in Campuswire was used during the model testing yielding positive results.

link: <https://campuswire.com/c/G3D46BBBA/feed/419>

* Irrelevant observations Following additional recommendations in the previous link, other type of irrelevant observations that were removed, corresponded to information that was already captured by other variables. A good example is the “Longitude” and “Latitude” variables. The rationale is that there were other variables that provided more interesting information about the neighborhood.

List of Variables Removed:

'PID', 'Street', 'Utilities', 'Condition\_2', 'Roof\_Matl', 'Heating', 'Pool\_QC', 'Misc\_Feature', 'Low\_Qual\_Fin\_SF', 'Pool\_Area', 'Longitude','Latitude'

Ultimately, testing showed that both Lasso and XGBoost models accuracy metrics improved after after removal of these irrelevant variables.

* Winsorization: The next step was to “Filter-out Outliers”, which are datapoints that differ significantly from other observations in the dataset. In order to accomplish this goal, a “winsorization” process was applied to some numerical variables. For example, let’s consider the variable “Lot\_Frontage”. The next plots show the data distribution before and after the winsorization process.



BEFORE

AFTER

Outliers

List of variables selected for “winsorization”:

"Lot\_Frontage", "Lot\_Area", "Mas\_Vnr\_Area", "BsmtFin\_SF\_2", "Bsmt\_Unf\_SF", "Total\_Bsmt\_SF", "Second\_Flr\_SF", 'First\_Flr\_SF', "Gr\_Liv\_Area", "Garage\_Area", "Wood\_Deck\_SF", "Open\_Porch\_SF", "Enclosed\_Porch", "Three\_season\_porch", "Screen\_Porch", "Misc\_Val"

* Categorical Variables handling: One-Hot-Encoding was used to transform each categorical value within a variable, into a new categorical column and assign a binary value of 1 or 0 to those columns. For this process, the dummyVars function form the Caret library was used.

After completing the dummyVars function, we observed an expansion on both the training and test matrix going from 70 columns to 300 columns.

* Matching transformed Training and Test matrix: For the Test datasets, one additional step was done after doing One-Hot-Encoding. The goal is to ensure both the Training matrix and the Test matrix had the same number variables and the same column order. This step was needed because the Test data set may not have the same values in some of the categorical variables, thus when the One-Hot-Encoding is done on the Test dataset, the resulting matrix will have some levels not observed in the training dataset, thus the fitting model cannot be used. In order to overcome this, the approach was two create a two dataframes using the R function setdiff(). The first dataframe had the name of the variables in the Train dataset but missing in the Test dataset. The second dataframe had the name of the variables in the Test dataset but missing in the Train dataset.

The procedure is as follow: all the missing variables in the Test dataset that are in the Train dataset were added into the Test dataset with values = 0. All the variables in the Test dataset that are missing in the Train Dataset were removed from the Test dataset. Lastly, the column order in the Test data set were rearranged to match the column order in the Train dataset.

**Linear Methods**

After completing all the data cleansing and pre-processing previously described, a linear Lass model, with cross-validation to select the best minimum lambda met the minimum benchmark for most of the test splits. However, there were some case where the results were not consistently meeting the performance targets. To improve the model, an additional step was done to select the significant variables from Lasso, and therefore remove the variables that were “zeroed” by the Lasso process, and used that list to fit a regression using Ridge. At this point, consistent performance results were observed.