# Predicting home prices
# with linear, lasso, ridge and random forest regression

A first capstone project for
Springboard's data science bootcamp

Mike Pierovich
February 26, 2020

# 1. Introduction

"What will my home sell for? And why?" These are basic questions asked by almost every homeowner.

In this capstone project, my hypothetical client is a website focused on helping homeowners. Specifically, the website wants to help homeowners:

- Make better decisions around buying and selling homes,
- Make smarter choices around remodeling and upgrading homes, and
- Have more-effective collaborations with real estate agents.

To support the client, I do the following:

- Explore a data set of sales prices and home features from one U.S. city over a 5-year period,
- Evaluate which home features--such as home size, quality, and location--had the most impact and influence on sales price,
- Develop models to predict the sales price of residential homes,
- Identify the best-performing model on my test data.
- Use the best-performing model to predict prices and prediction intervals at a 95% confidence-level with respect to the test set used.

# 2. Data acquisition

I use data from the Kaggle competition, “House Prices: Advanced Regression Techniques.” The dataset is called the “Ames Housing” dataset. The Ames Housing training dataset includes:

- About 1,500 observations of the sales of residential properties in the city of Ames, Iowa,
- About 80 features that describe different aspects of each home. These features include, for example, the number of rooms, number of bathrooms, year built, the size of the house, etc.
- The sales price of each home as well as the year and month sold.

# 3. Initial cleaning

This data is in a clean state as it comes from Kaggle. The data is "tidy." Rows are observations. Each row has a non-null target variable. Columns are variables and well labeled. There is a well-documented data dictionary.

I do the following to further clean the data:

- Create an index from existing IDs,
- Turn date- and time-based features into date times,
- Resolve null values for the non-categorical features. For these features, null often means zero values.
- Convert ordered categorical features into integers. There are many categorical features with string-based values that can be represented by integers. Often, these features categorize quality, condition, or amount. Examples of such categories are exterior quality, garage condition, or the amount of slope to a lot.
- Clean up null values in categorical features. As an initial approach, I leave most categorical nulls as NaN and clean only those where there were two or more conflicting categories that conceptually mean null. For example, if there is a column with both “NaN” and “None,” I made them all “None.” But later, I realize that I need to recode NaN into an explicit category (often, a “None” or “No” or something similar, for example).

# 4. Data exploration

To better understand the dataset, I conduct exploratory data analysis (EDA) on the target variable and predictive features.

## 4.1. EDA on the target variable

I explore the target variable--the sales price of residential homes. I look at the following:

- **Central tendency:** The mean sales price is about $180k. The median is about $160k. The minimum sales price is $35k.  The maximum sales price is $750k.

- **Distribution:** The distribution of sales price isn't normal. Rather, it is positively skewed with a longer upper tail as shown in figure 1. The distribution has a hint of bi-modality--with one peak around $150k and a second, smaller peak at $200K. The distribution of the log of price looks significantly more normal as shown in figure 2.

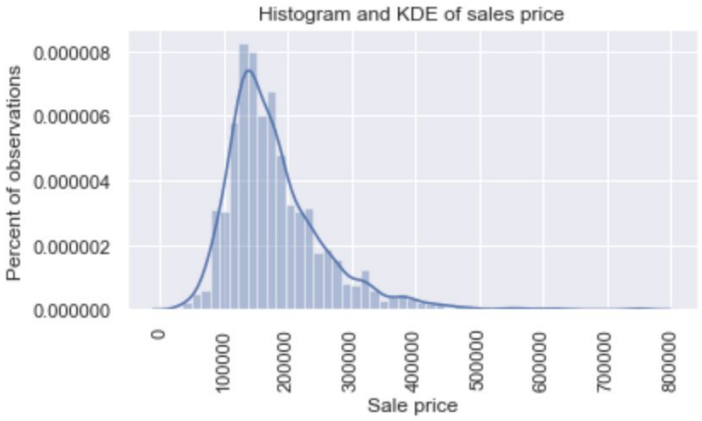


Figure 1: Histograms and KDE of the sales price

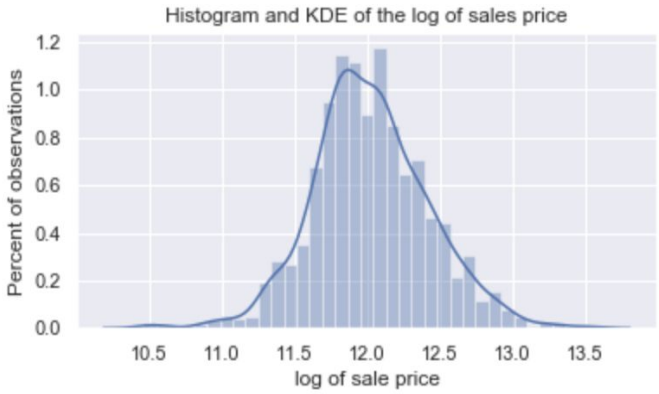


Figure 2:Histograms and KDE of the log of the sales price

- **Outliers:** There are outliers in the sales price. These are the rare, but very expensive homes. Outliers are illustrated in figures 3 and 4. There are 22 sales three times beyond the standard deviation of the sales price. There are eight sales located at a distance greater or equal to one and a half times the interquartile range (IRQ) above the 75th percentile (i.e., Tukey inner fences). I use this as my definition of an outlier. There are no points at a distance greater or equal to three times the interquartile range above the 75th percentile (i.e., Tukey outer fences).

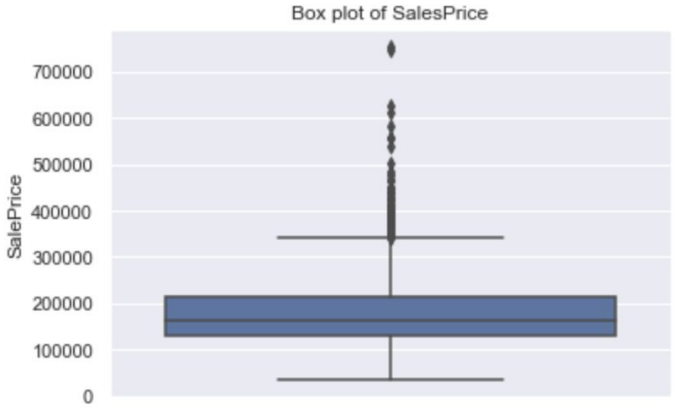


Figure 3: Box plot of sales price

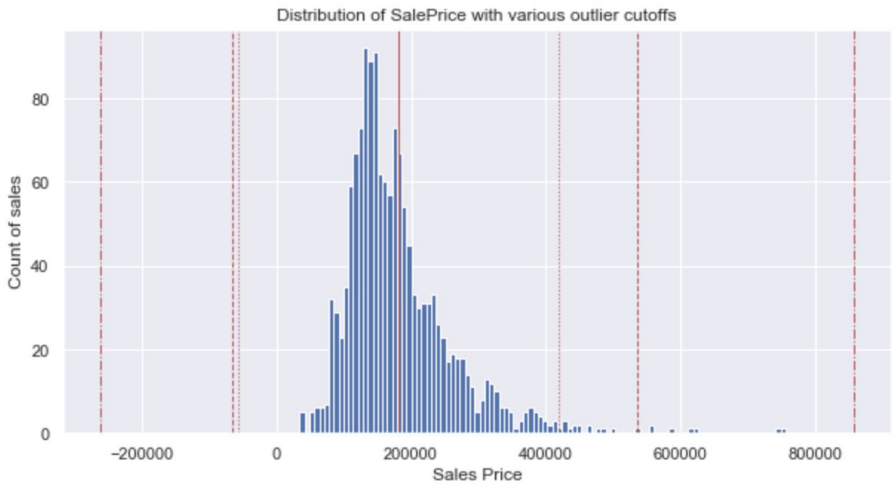


Figure 4: Distribution of sales price with outliers cutoffs as red-vertical lines at 3 x standard deviation (red dotted), 1.5 x IRQ (red dashed) and 3 x IRQ (red dashed-dotted)

- **Sales per year:** The dataset covers five years from 2006 to 2010. Sales per year are shown in figure 5. The year 2010 is a partial year (i.e., sales for only the first six and a half months of the year). There are about 300 sales in each full year.

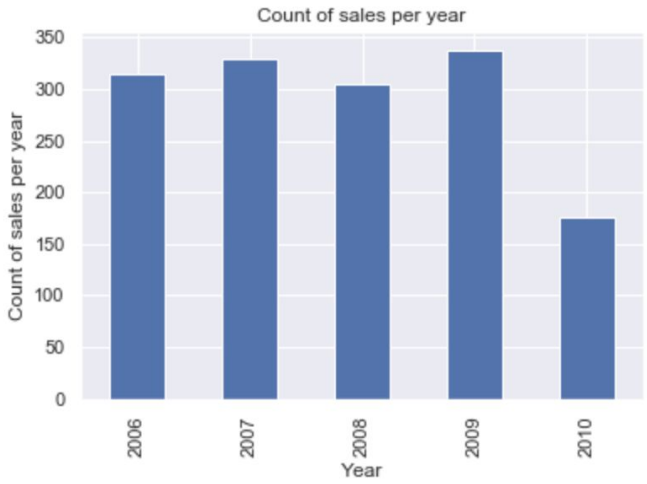


Figure 5: Count of sales by year

- **Sales per calendar month:** The typical calendar month has about 23 sales (median) and 27 sales (mean). But there are big variations from month to month. This is shown in figure 6.


Figure 6: Count of sales by calendar month

- **Seasonality:** Sales are highly seasonal. The biggest months (i.e., May, June, and July) have four to five times the volume of the smallest months (i.e., Dec and Jan). Sorting by seasons makes this even more clear. The summer has four times the volume of sales as the winter, which is illustrated in figure 7. Clearly, more people buy and sell homes in the summer.

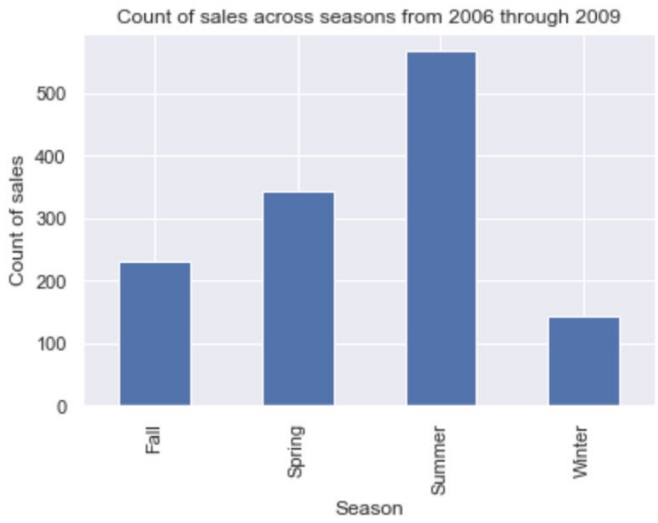


Figure 7: Count of sales by season for the four full years from 2006 through 2009

- **Sales price by year:** For this period, there is no clear trend of year-over-year sales prices increases, which isn't what I expected. Prices have been ever-so-slightly dropping, and price variability might be increasing. This is shown in figure 8. It's worth noting that these years coincide with the "Great Recession" in the United States, which included the end of a housing bubble, bank troubles caused by real estate derivatives, and other macroeconomic woes.

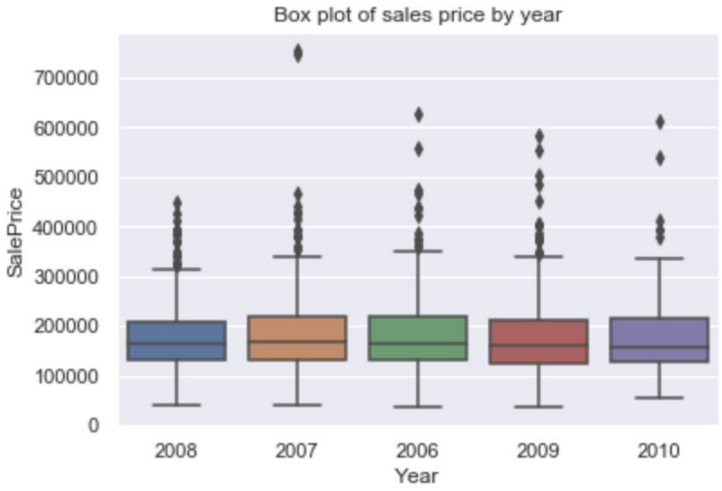


Figure 8: Box plots of sales price by year

## 4.2. EDA on predictive features

Next, I set out to understand more about my 79 predictive features. Specifically, I want to understand which of these features have independent, explanatory power with respect to my target variable.

To do this, I separate my non-categorical and categorical variables. Then, I do the following:

- Decide to use a Person's r at 30% as a floor for what I consider to be sufficiently well-correlated with my target variable (i.e, the correlation floor) and p-values of 5% as cutoffs,
- Calculate Person's r, f-stats, and p-values for each feature,
- Visually inspect those features that meet the cut-off,
- Decided to use a Person's r at 50% as a ceiling for what I consider to sufficiently non-correlation between features (aka, a cross-correlation ceiling),
- Calculate cross-correlation between features as needed,
- Visually inspect cross-correlations via a heat map as needed, and
- Identify features that hit both the floor and the ceilings.

Here are my findings:

- **Correlation of non-categorical features with sales price**: Of the 54 non-categorical features, only 24 meet the correlation floor of 30%. All have very small p-values. All are listed in table 9 below.

| Non-categorical features | R-squared | P value | Reject null |
|---|---|---|---|
| OverallQual | 0.791 | 0.000 | Yes |
| GrLivArea | 0.709 | 0.000 | Yes |
| ExterQual | 0.683 | 0.000 | Yes |
| KitchenQual | 0.660 | 0.000 | Yes |
| GarageCars | 0.640 | 0.000 | Yes |
| GarageArea | 0.623 | 0.000 | Yes |
| TotalBsmtSF | 0.614 | 0.000 | Yes |
| 1stFlrSF | 0.606 | 0.000 | Yes |
| BsmtQual | 0.585 | 0.000 | Yes |
| FullBath | 0.561 | 0.000 | Yes |
| TotRmsAbvGrd | 0.534 | 0.000 | Yes |
| YearBuilt | 0.523 | 0.000 | Yes |
| FireplaceQu | 0.520 | 0.000 | Yes |

| | | | |
|---|---|---|---|
| GarageYrBlt | 0.508 | 0.000 | Yes |
| YearRemodAdd | 0.507 | 0.000 | Yes |
| MasVnrArea | 0.473 | 0.000 | Yes |
| Fireplaces | 0.467 | 0.000 | Yes |
| HeatingQC | 0.428 | 0.000 | Yes |
| BsmtFinSF1 | 0.386 | 0.000 | Yes |
| BsmtExposure | 0.362 | 0.000 | Yes |
| WoodDeckSF | 0.324 | 0.000 | Yes |
| 2ndFlrSF | 0.319 | 0.000 | Yes |
| OpenPorchSF | 0.316 | 0.000 | Yes |
| BsmtFinType1 | 0.305 | 0.000 | Yes |

Table 9: 24 non-categorical features with R-squared, p-values and conclusions, as calculated against the sales price

- **Scatter plots against sales price:** Figures 10 and 11 are example scatter plots of the two well-correlated features with the sales price. The positive correlation is readily apparent.

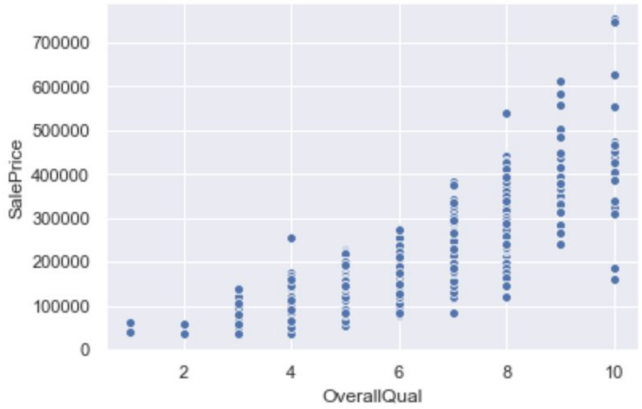


Figure 10: Scatter plot of overall quality (a rating of the overall material and finish of the house) against the sales price

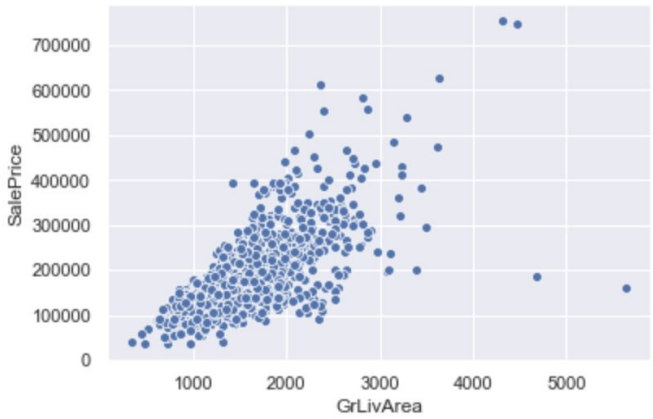


Figure 11: Scatter plot of GrLivArea (above ground living area square feet) against sales price

- **Heatmap of cross correlation between non-categorical variables:** Figure 12 is a heatmap of cross-correlation metrics between these 24 variables. The high cross-correlation is readily apparent. The cross-correlated features include measures of quality (e.g., kitchen quality and exterior quality); measures of space (e.g., garage square footage and the number of cars); and other measures (e.g., the number of rooms and number of bathrooms).

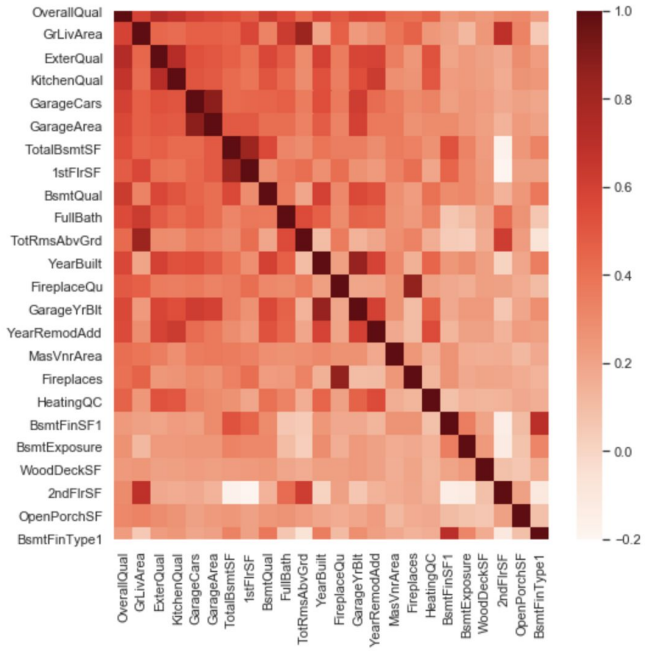


Figure 12: Heatmap of cross-correlation metrics among the top 24 features

- **Final non-categorical features:** After screen for the highest Pearson's r that lack cross-correlation of 50% or more to any other features, ten remain. They are listed in table 13 below.

| Non-categorical features |
|---|
| OverallQual |
| 1stFlrSF |
| TotRmsAbvGrd |
| FireplaceQu |
| MasVnrArea |
| HeatingQC |
| BsmtFinSF1 |
| BsmtExposure |
| WoodDeckSF |
| OpenPorchSF |

Table 13: List of final non-categorical variables
that meet correlation floor and cross-correlation ceiling

- **Correlation of non-categorical features with sales price**: Of the 25 categorical features, only 2 meet the correlation floor of 30%. All have very small p-values. All have more than 2 factors. All are listed in table 14 below.

| Categorical features | R-squared | P value | Reject null | No. factors |
|---|---|---|---|---|
| Neighborhood | 0.550 | 0.000 | Yes | 25 |
| GarageFinish | 0.310 | 0.000 | Yes | 4 |

Table 14: Two categorical features with R-squared, p-values, conclusions
and number of factors, all as calculated against the sales price

- **Box plots of categorical features against sales price:** Figures 15 and 16 are box plots of the two best-correlated features with the sales price. The box plots are broken out by factors. The plots do give hints that there is some correlation between IRQ of price by factors.

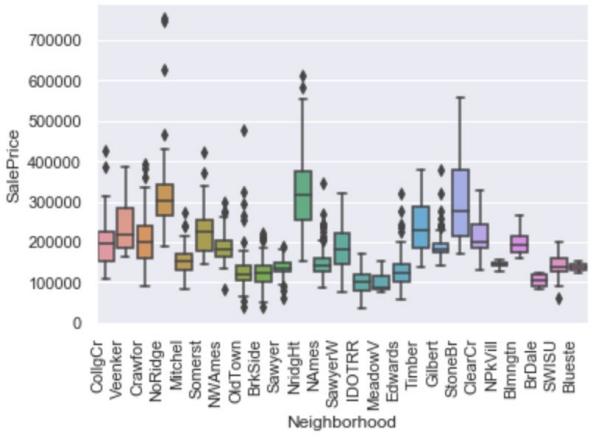


Figure 15: Box plot of Neighborhoods (e.g, physical locations within Ames city limits) against the sales price

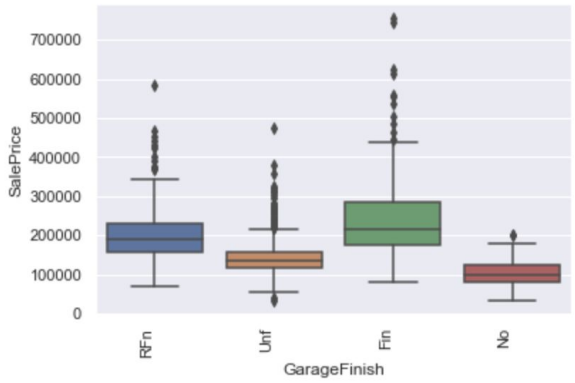


Figure 16: Box plot of Garage Finish (i.e., the interior finish of the garage) against the sales price

- **Final categorical features:** I decide to use both categorical features--Neighborhood and GarageFinish--in my model. They are listed in table 17.

| Categorical features |
|---|
| Neighborhood |
| GarageFinish |

Table 17: List of final non-categorical variables that meet correlation floor

- **Final features:** All in all, I decide on 12 features, which include ten non-categorical features and two categorical features, as described above.

Upon finishing my EDA, I have an understanding of my target variable. Also, I have identified which variables are well-correlated with that target and which are reasonably independent of each other. Now, I am ready to start modeling.

# 5. Data pre-processing

To prepare my data for modeling, I do the following pre-processing:

- Drop the eight outliers that were 1.5 IRQ above the 75th percentile. Here is my thinking. The observations might throw off my models, and there are very few of these outliers (so I am not losing much information value).
- Apply log transformation to the target variable, and
- Convert categorical variables to numerical variables with one-hot encoding.

When done, I have 37 features to use for modeling.

# 6. Basic model

To start, I use linear regression as my basic model. I choose this model because:

- This is a supervised learning problem with a continuous dependent variable (i.e., sale price) for each observation.
- I am interested in interpretability. That is, I not only want to predict prices but also want to help users understand what features drive the predictions.

My general approach is to:

- Start with Ordinary Least Squares regression as a baseline,
- Use Ridge and Lasso regression if I find overfitting,
- Score with r-squared (R2), root-mean-square error (RMSE), and mean absolute error (MAE),
- Split my data into training and test sets on a 75%/25% basis, and
- Keep this split across all models to help me compare performance.

Below are more details on these steps.

**Evaluate a simple model:**

- I tune the (few) parameters of linear regression and find the best performance associated with: feature normalization turned off and y-intercept calculated.
- I score this model on the training and test sets.
- I do not see evidence of overfitting as errors in the test set were slightly less that than those of the training set.
- I find that the model's performance is better than I expect, which is set forth in table 18 below. R2 is in the mid-80% while the highest R2 of each variable alone, for example, is in the 70's, with most in the 60's. R2 only increases with additional variables, of course. But still, this goes up more than I expect.

| Model | Data | RMSE | MAE | R2 |
|---|---|---|---|---|
| Linear regression - baseline | Train | 0.1548 | 0.1122 | 0.8494 |
| Linear regression - baseline | Test | 0.1509 | 0.1095 | 0.8578 |

Table 18: Performance results of the base model on train and test data

**Evaluate whether this model's success was luck of the draw:**

- I wonder if the random draw behind my test/train split impacted these results. I compare the error of this draw versus that of 999 others. All samples are created from the same 75/25 split that I use for the baseline model. The only difference is the random seed--that is, the draw of observations into the train or test split.
- I find that this model's mean-squared error (MSE) was within the IRQ of the 1,000 draws.
- This gives me the intuition that my particular draw wasn't an outlier--although I could do a more formal test to fully validate this.

**Evaluate whether normalizing features reduces performance:**

- My baseline model does not normalize the predictor variables. Normalizing these predictors--i.e., subtracting the mean and dividing by the standard deviation--would make it easier to compare across predictors. So, I want to understand the performance cost of enabling normalization.
- I find that a normalized model performs the same as the model without normalization.
- I decide to normalize my features for improved interpretability.

**Compare whether Ridge or Lasso performs better:**

- I don't find overfitting so I don't need to switch to Ridge or Lasso. Still, I score the models.
- I find that the linear model gives the best results by a small margin. Followed by Lasso. Followed by Ridge. These results are listed in table 19.
- I visually inspect Lasso's regression coefficients, which highlight a number of less important features. These coefficients are visualized on table 20.
- I make the decision to continue to use the linear model with normalized data.

| Model | Data | RMSE | MAE | R2 |
|---|---|---|---|---|
| Linear regression - baseline - normalized | Test | 0.1509 | 0.1095 | 0.8578 |
| Lasso regression | Test | 0.1528 | 0.1098 | 0.8543 |
| Ridge regression | Test | 0.1534 | 0.1112 | 0.8530 |

Table 19: Performance results of Linear, Lasso, and Ridge on test data, sorted by R2

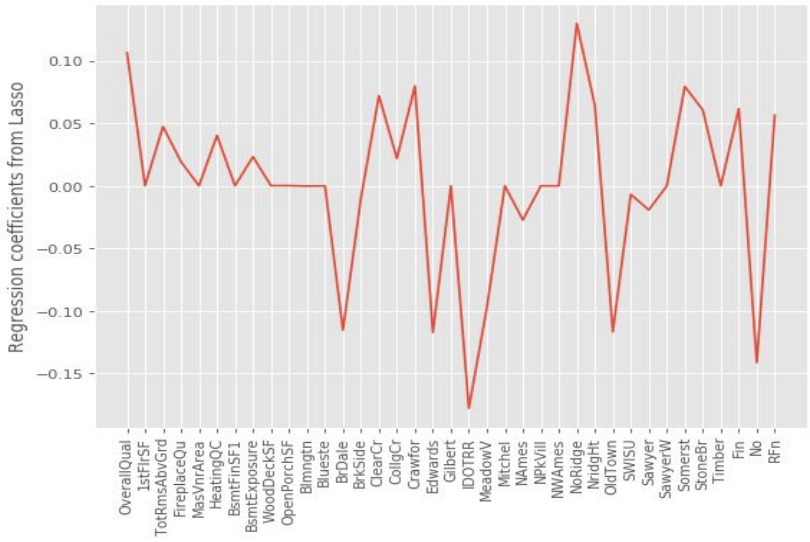


Figure 20: Regression coefficients in the Lasso regression model

**Use linear coefficients to better understand trade-offs between features:**

- I inspect the various regression coefficients from my linear model with normalized features. The head and tail of the list of coefficients for my 37 features are listed in table 21 below.
- These coefficients tell how much an increase in one of the predictors impacts the log of the sales price. For example, an increase in the overall quality measure by 1 increased the log sales price by about 0.097.
- As all predictors had been normalized, the coefficients are on the same scale. This gives me a way to understand the relative contribution of changes in coefficients to the log of the sales price. For example, the same relative improvement in overall quality feature increases the value by more than the same relative improvement in total rooms above ground.
- These coefficients also tell me about the direction of changes. An example can be found in garage finishes. The default in my model is an unfinished garage. Adding a full finish to the garage (aka, the highest quality) adds the most value. Adding a rough finish to the garage (aka, medium quality) adds some value but not as much. Having no garage (aka, eliminating the garage) destroys value.

| Feature | Coefficient | Type of feature |
|---|---|---|
| NoRidge | 0.0971 | cat_neigborhood |
| OverallQual | 0.0927 | noncat |
| Fin | 0.0604 | cat_garage_finish |
| TotRmsAbvGrd | 0.0486 | noncat |
| RFn | 0.0474 | cat_garage_finish |
| ... | ... | ... |
| Edwards | -0.2557 | cat_neigborhood |
| Blueste | -0.2705 | cat_neigborhood |
| MeadowV | -0.3313 | cat_neigborhood |
| BrDale | -0.3405 | cat_neigborhood |
| IDOTRR | -0.3504 | cat_neigborhood |

Table 21: Head and tail of the list of regression coefficients
from the linear model with normalizations, sorted by coefficient value

**Use the model to create prediction intervals:**

- I select a home at random from the test set. It has 4 rooms, 936 square feet on the first floor, and a quality index of 5, plus it's located in the neighborhood of North Ames and has a finished garage. All in all, it is a rather average home, maybe with slightly smaller-than-average rooms.
- The model predicts a sales price of about $124,000 (against the actual sales price of $129,000).
- I calculate a 95% prediction interval using the actual residuals at the 2.5% percentile and 97.5% percentile.
- This gives me a range between $96,000 and $169,00.  I might use this range to account for a best- and worst-case scenario.  That is, almost all of the time, the home's sales price should be between these two prices.

# 7. Extended model

I use random forest regressors as my extended model. I choose this algorithm because of the model's strength with regression,  because it is different than regression (e.g., a non-parametric, ensembled approach) and because of random forest's general utility.

My approach to random forest regression involves the following:

- Maintain my prior test split,
- Score a baseline model of mainly default parameters as defined by scikit-learn RandomForestRegressor class,
- Continue to score with RMSE, MAE, and R2,
- Add an out-of-bag (OOB) score,
- Tune based on key parameters,
- Tune based on feature importance by assessing the various features and scoring the model using different combinations of features, and
- Find the best performing random-forest-regression model.

Below are more details on these steps.

**Score a baseline model:**

- For my baseline random forest model, I start with the default parameters as defined by scikit-learn RandomForestRegressor class.
- Then, I change two non-default parameters.
    - I set the number of estimators to the minimum plus 1, which avoids an annoying warning.
    - I turn on bootstrapping.  Bootstrapping is needed to generate OOB scores.  This most certainly makes sense as OOB score is calculated off the samples not used in the bootstrap.  But it's

not well documented by scikit-learn. I discover this only when I encounter an error: “ValueError: Out of bag estimation only available if bootstrap=True”.

- The baseline scores are in table 22 below.
- All in all, the baseline model for random forest regression has a strong performance. RMSE and MAE are low, and it isn’t surprising that RMSE is higher than MAE as my data has outliers. With OOB at 80%, the model is strong with out-of-bag samples. With R2 at 83%, the model has strong explanatory power.

| Model | Data | RMSE | MAE | R2 | OOB |
|---|---|---|---|---|---|
| RF regression - baseline | Test | 0.1625 | 0.1150 | 0.8352 | 0.8030 |

Table 22: Performance of baseline random forest regression

**Tune model using parameters:**

- I tune this model in two steps. First, I tune the number of estimators (e.g, the number of trees in the forest). Second, I tune across a grid of five other top-of-mind parameters.
- In tuning for the number of estimators, the best accuracy score is associated with the highest number of estimators--10,000. This leads me to conclude that the model does better with more estimators. However, to avoid having to wait 20+ minutes each time I run my notebook, I decide to limit my model to 1,000.
- For additional tuning, I grid search across: bootstrapping on/off, maximum tree depth, maximum features per tree, minimum samples per leaf, and minimum samples on each split. I use the highest-scoring combination of parameters. However, I do not use the recommended bootstrapping off as I wanted to leverage OOB scores.
- The parameters that I use for my tuned model are listed in table 23.

| Parameter | Value |
|---|---|
| Number of estimators | 1,000 |
| Bootstrapping on/off | On |
| Maximum tree depth | 25 |
| Maximum features per tree | 10 |
| Minimum samples per leaf | 2 |
| Minimum samples on split | 2 |
| Out of bag scoring on/off | On |

Table 23: Parameters for tuned random forest regression model

**Score tuned model:**

- Next, I score the tuned model and compare it to the baseline model.
- The results of the baseline and the tuned model are in table 24 below.
- The tuned model performs better. It has less error with a smaller RMSE. Its OOB score is 3% higher at about 83%. It has more explanatory power--with an R2 of about 2% more.
- To me, the tuned model's improved performance makes sense. Metaphorically speaking, my tuning magnifies the "wisdom of the crowds" aspect of this ensemble.
    - With the tuned model, I sample a much bigger crowd. The number of estimators (aka, trees) went up by 50x. From about 20 to 1,000.
    - With the tuned model, I ensure my crowd was "less expert" (or less overfit). That is, I use more-simplified trees by capping the unlimited depth at 25-levels and by increasing the leaf size from 1 to 2 samples.

| Model | Data | RMSE | MAE | R2 | OOB |
|---|---|---|---|---|---|
| RF regression - baseline | Test | 0.1625 | 0.1150 | 0.8352 | 0.8030 |
| RF regression - tuned | Test | 0.1543 | 0.1084 | 0.8514 | 0.8354 |

Table 24: Performance of baseline and tuned random forest regression models

**Assess feature importance:**

- I assess the importance of the features to the random forest model.
- Figure 25 below visualizes the feature importance.
- I sort features into 4 groups.
    - High-importance features: 3 features with a feature importance score above .1,
    - Medium-importance features: 4 features with a score above .05 and below .01,
    - Low-importance features: 20 features with a score above .001 and below .05, and
    - Very-low-importance features: The remaining 10 features with a score below .001.
- Table 26 lists the features by importance and by group.
- The high- and medium-importance features are 7 of the 9 non-categorical features that I originally selected. And they are ordered in almost the same order as their ranking by univariate R2.
- The low-importance features are mostly the enumerated types from my non-categorical variables. Of these 20 low-importance features, most (14) are specific neighborhoods, presumably those which gave me more information gain. Some (3) are categories of garage finish--the other categorical variable.

- The 10 very-low-importance features are all of the same type. All are one-hot-encoded neighborhoods. Presumably, they all lack a lot of unique information that contributes to the price.

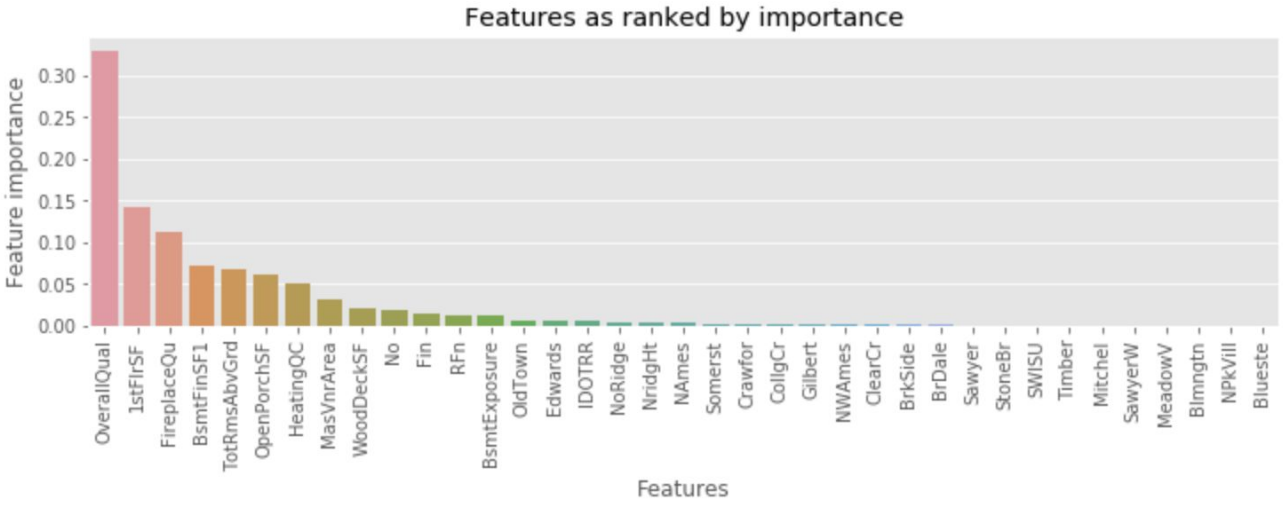


Figure 25: Features in the random forest tuned model, as ranked by importance

| Feature | Importance | Group |
|---|---|---|
| OverallQual | 0.3299 | Hi |
| 1stFlrSF | 0.1413 | Hi |
| FireplaceQu | 0.1123 | Hi |
| BsmtFinSF1 | 0.0722 | Medium |
| TotRmsAbvGrd | 0.0685 | Medium |
| OpenPorchSF | 0.0620 | Medium |
| HeatingQC | 0.0503 | Medium |
| MasVnrArea | 0.0324 | Low |
| WoodDeckSF | 0.0216 | Low |
| No | 0.0182 | Low |
| Fin | 0.0155 | Low |
| RFn | 0.0128 | Low |
| BsmtExposure | 0.0125 | Low |
| OldTown | 0.0071 | Low |
| Edwards | 0.0059 | Low |
| IDOTRR | 0.0055 | Low |

| | | |
|---|---|---|
| NoRidge | 0.0047 | Low |
| NridgHt | 0.0044 | Low |
| NAmes | 0.0044 | Low |
| Somerst | 0.0028 | Low |
| Crawfor | 0.0024 | Low |
| CollgCr | 0.0019 | Low |
| Gilbert | 0.0017 | Low |
| NWAmes | 0.0016 | Low |
| ClearCr | 0.0014 | Low |
| BrkSide | 0.0013 | Low |
| BrDale | 0.0012 | Low |
| Sawyer | 0.0009 | Very low |
| StoneBr | 0.0006 | Very low |
| SWISU | 0.0005 | Very low |
| Timber | 0.0005 | Very low |
| Mitchel | 0.0005 | Very low |
| SawyerW | 0.0004 | Very low |
| MeadowV | 0.0004 | Very low |
| Blmngtn | 0.0001 | Very low |
| NPkVill | 0.0001 | Very low |
| Blueste | 0.0000 | Very low |

Table 26: List of features with importance and grouping by importance

**Score models using different feature sets:**

- I score my tuned model using 4 different features sets:
    - All features,
    - High-, medium- and low-importance features (i.e., dropping the very-low-importance features),
    - High- and medium-importance features (i.e., dropping the low- and very-low-importance features), and
    - High-importance features only (i.e., dropping the medium-, low- and very-low-importance features).
- The results are in table 27.
- The best-performing model is that with high-medium-low-importance features. Here's why:

    - The model with all features has the highest OOB score, followed closely by the model with high-medium-low features.
    - The model with high-medium-low features has the lowest RMSE and MAE, followed closely by the model with all features.
    - The model with high-medium-low features has the most explanatory power (R2), followed closely by the model with all-features.
    - Basically, the “high-medium-low” model and the “all” model are better than those with “only high” and “only high-and-medium”.
    - As between “high-medium-low” and “all”, I choose “high-medium-low.” I am willing to trade a small loss on OOB score for small gain on accuracy and explanatory power
- As I understand it, the model improves by eliminating the very low features, which, presumably, were about 10 neighborhoods that didn’t have much information value as compared to the default neighborhood.

| Model | Data | RMSE | MAE | R2 | OOB |
|---|---|---|---|---|---|
| RF regression - tuned - high, med, low | Test | 0.1535 | 0.1079 | 0.8529 | 0.8348 |
| RF regression - tuned - all features | Test | 0.1543 | 0.1084 | 0.8514 | 0.8354 |
| RF regression - tuned - high, med | Test | 0.1676 | 0.1203 | 0.8248 | 0.8006 |
| RF regression - tuned - high | Test | 0.1873 | 0.1390 | 0.7810 | 0.6961 |

Table 27: Performance of tuned model on different feature sets, including all features; high-medium-low-importance features, high-medium-importance features, and high-importance features

**Calculate a prediction interval from the tuned model of the “high-medium-low” feature.**

- Here, I calculate a prediction interval off the residuals from the tuned random forest model using “high-medium-low” features.
- I use the same property that I used for the linear regression prediction interval to allow me to compare the prediction intervals across the models.
- This model predicts a sales price of about $119,000 (against the actual sales price of $129,000).
- The 95% prediction interval is between about $89,000 and $158,000.

**Visualize a single tree:**

- As part of this exercise, I visualize a randomly selected tree that I used in an early baseline random forest model. The larger tree is illustrated in figure 28. The first node of the tree is highlighted in figure 29.
- In looking at the entire tree, I see the depth of the tree as well as the combinations of features that lead to lower-value predictions (generally, more on the left) and the higher-value predictions (generally, more on the right)
- In looking at the top-level individual node, I see:
    - The specific feature and the value used in the node. Here, it's "fireplace quality."
    - The node's MSE. In the first node, it's .1.
    - The count of samples before and after the split. In this first node, it's 700 samples, which gets split into 341 on the "True" branch and 359 on the "False" branch. Plus, the high-information gain on that first split is highlighted by the fact the node is splitting the sample almost 50/50.
    - The predicted value of sales price at the time of the split. In this node, it's 12.0 as the log of the sales price.

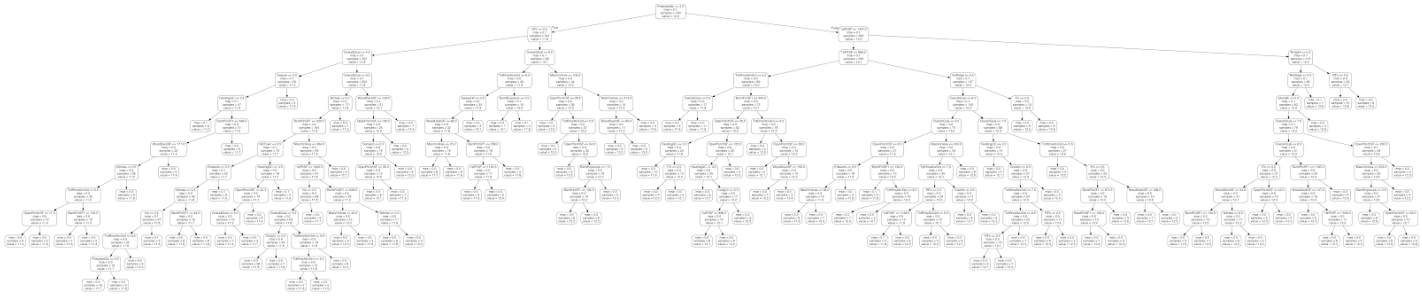

Figure 28: A visualization of a randomly-selected tree
from an early version of my baseline random forest regressor

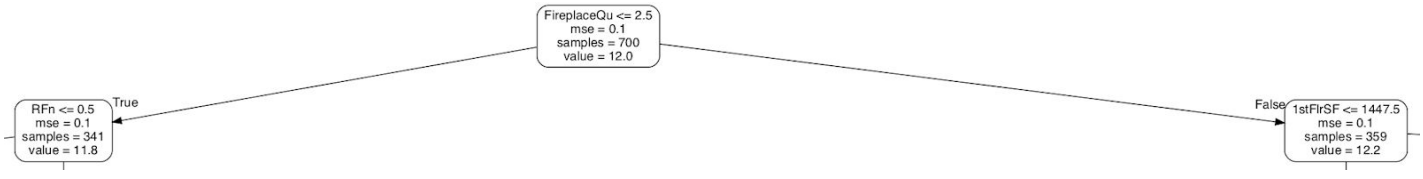


Figure 29: A visualization of the top-level nodes from the randomly-selected tree
from an early version of my baseline random forest regressor

# 8. Conclusions

Here are my conclusions about the relative performance of my strongest linear regression model and my strongest random forest regression model.

- The normalized linear regression model is the best model. The performance results are in table 30.
- The models are split on errors. Random forest regression has a bit better RMSE. Linear regression has a bit better MAE.
- The linear regression model is a bit better on R2.
- I select linear regression for that bit of R2 improvement.

| Model | Data | RMSE | MAE | R2 | OOB |
|---|---|---|---|---|---|
| Linear regression - baseline - normalized | Test | 0.1509 | 0.1095 | 0.8578 | N/A |
| RF regression - tuned - high, med, low | Test | 0.1535 | 0.1079 | 0.8529 | 0.8348 |

Table 30: Performance of best linear model and best random forest model

- I also compare predictions from these two models. The details are in table 31 below.
- There is a difference of about $5,000 or 4% between the two predictions.
- The random forest model's prediction is a bit closer to the actual value.
- The range of the random forest's interval is a bit narrower at $68,000 as compared to the linear model's interval, which is $74,000.
- In this specific case, maybe the prediction interval of the random forest is a bit better.
- Of course, the exercise of comparing one individual prediction from each model isn't all that informative. Still, I find it to be useful context.

| Model | Lower bound | Prediction | Upper bound | Actual |
|---|---|---|---|---|
| Linear regression - baseline - normalized | $89,496 | $119,227 | $157,641 | $129,000 |
| RF regression - tuned - high, med, low | $95,483 | $123,635 | $169,181 | $129,000 |

Table 31: Performance of best linear model and best random forest model

Here are my conclusions regarding features:

- I benefit from setting a rigorous screen of whether to include categorical and non-categorical features in my models. As previously stated, my screen was:
    - First, an R2 of at least 30% when individually regressed against our dependent variable and a p-value lower than 5%, and
    - Second, a cross-correlation measure of no more than 50% against any other variables of the same type.
- This cut our variables down from around 80 to 12.
- The lack of unnecessary variables was borne out when tuning the random forest regression based on features. The model only (slightly) improved when I eliminated a few factors from one categorical feature--that is, I didn't need to do any major feature adjustments.

# 9. Recommendations

I offer the client two types of recommendations.

First, I recommend that the client build a "minimum viable product"(MVP) using the best model for the one city where we're got available data. This MVP has enough information to give:

- A predicted price,
- A confidence interval,
- For a specific property, and
- For all properties in Ames, Iowa.

Our results are good enough to start collecting evidence on whether our predictions actually meet the needs of our consumers to better understand sales price.

Second, I recommend that I work with the client to improve our data, features, and models. Specifically:

- Obtain data for locations other than Ames, Iowa.
    - Our data is sourced from only one city. I should explore how to obtain similar data for other locations.
- Obtain additional sources of data:
    - Adding macro-economic features (like, like interest rates, GDP growth, wage growth, etc.), and
    - Adding banking features (like real estate loans, loan rates, etc.).
- Obtain historical data
- Improve the features by:
    - Exploring whether dimensionality reduction lets me get some "signal" from many overlapping features

    - Experimenting on whether I can engineer new features from existing features
- Improve the linear regression models by:
    - Better analyzing whether the assumptions of linear regression apply, and
    - Systematically selecting features via techniques like stepwise selection, forward selection, backward elimination, etc.
- Explore other models by:
    - Looking at other regression models, such as polynomial regression and ElasticNet regression,
    - Looking at other nonparametric models, such as gradient boosting, and other ensemble techniques, and
    - Looking at Bayesian approaches that might help overcome the limited data.