The Kaggle Zillow Prize - Improving the Zestimate

Shyamalee Ramesh

October 21, 2017

Research Question :

Zillow has 'Zestimates', that are estimated home values based on hundreds of data points on a property. Given the features of properties in three counties in California, would you be able to further reduce the Mean Absolute Error from what Zillow has it at now?

Introduction :

The Zestimate is Zillow's estimated market value for a home, computed using a proprietary formula from publicly available and user submitted data. It is intended as an useful starting point for the user to determine an unbiased assessment of what a home would be worth in the current market. According to Zillow, the Zestimate's accuracy depends on location and availability of the data in the specific neighborhood. The Kaggle competition on Zillow's Home Value Prediction challenges participants to improve the algorithm that changed the world of real estate.[1] The participants are asked to develop an algorithm that will predict the future sale prices of homes. The submissions are evaluated on Mean Absolute Error between the predicted log error and the actual log error. The log error is defined as:

$$logerror = log(Zestimate) - log(SalePrice)$$

The Mean Absolute Error is given by:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y_i}|$$

Zillow has provided us with data including multiple features for nearly 3 million properties in three counties, Los Angeles, Orange and Ventura in California. The participants have to predict the Mean Absolute Error between the actual Sale Price and the Zestimate at six different points in time - October 2016, November 2016, December 2016, October 2017, November 2017 and December 2017 for every unique property. We also get the data about the log error between Zestimate and the actual Sale Prices for all the transactions that occured in 2016; if a transaction did not occur for a property during that period of time, that property is not included in the computation of Mean Absolute Error.

Exploratory Data Analysis

The dataset has features for nearly 3 million properties (2,985,217) but not all the properties have transaction information recorded; they have either not been sold in this time period or the information has not been captured. We include only those properties that had recorded transaction information also available in our analysis (90,275 properties). Exploratory analysis was performed

[1] Zillow's Zestimate Page, (https://www.zillow.com/zestimate/)

by examining all the variables and their distrbutions. [2]We can see that a lot of variables have a significant percentage of missing values; the following graph plots the degree of missingness of all the feature variables in properties (Figure 1) [3].

The purpose of the Exploratory Data Analysis is to figure out how our data looks, come up with visualizations that would help us understand how to proceed with our prediction problem. [4]. I made a correlation plot of all the numeric features - none of the numeric variables were correlated well with the log error (Figure 2). All the tax variables were strongly correlated with each other, so were the room count variables. When I plotted the distribution of log error over time, I could see that the Zestimate predicts the value of property better in more recent times (Figure 3).

I can also check the distribution of logerror, the response variable. I would like to check if the data appears normally distributed. It does look quite normally distributed around 0 (Figure 4). The Normality assumption is an important step for maintaining performance; as our data looks normal, a random sample of test data would have the same distribution as the entire data.

Dealing with Missing Values

Like we see from the above graph, the dataset has a lot of missing values and I used the following ways to impute missing data [5]:

- Variables with over ninety percent missing data and no feasible way to impute the correct value were discarded
- If variables related to the same or very similar information, only one of the variables were retained if they were highly correlated. Example: FIPS and Region ID County
- For variables with True/ NA's, make the TRUE values 1 and NA's 0 (ex: hashottuborspa)
- Make NA's 0 for count variables, such as bathroom count, pool count etc.
- If taxdeliquency year is present, but flag is 0/NA, make the flag 1
- Similarly, if count variables are greater than zero, but the flag is 0, make them 1
- Write a function to compute mode, impute missing values of a few numeric variables as the mode value (ex. FIPS)
- For variables such as tax value, impute the missing values as mean amount
- Change NA's across type variables to the code that specifies "other" (ex:AirconditioningtypeID)
- For a field like basementsqft, if values are NA, change them to 0, i.e. no Basement
- Impute a few variables by random sampling
- Check the total missingness now - none of the variables have missing values

Machine Learning Models and Prediction

1. Linear Regression

The first model I tried was a basic multiple linear regression with all the variables and the logerror as the response variable. The MLR model attempts to model the relationship between the features

[2]Hadley Wickham, Romain Francois, Lionel Henry and Kirill Müller (2017). dplyr: A Grammar of Data Manipulation. R package version 0.7.4. https://CRAN.R-project.org/package=dplyr

[3]H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

[4]Kernel on EDA in Zillow's Rent Estimate - Kaggle(https://www.kaggle.com/philippsp/exploratory-analysis-zillow)

[5]Frank E Harrell Jr, with contributions from Charles Dupont and many others. (2017). Hmisc: Harrell Miscellaneous. R package version 4.0-3. https://CRAN.R-project.org/package=Hmisc

and the response variable by fitting a linear equation to the observed data.

$$\mu_y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + ... + \beta_n X_n$$

where β_1 is the airconditioning type ID, β_2 is the basement square feet and so on including all the variables. The mean absolute error from this linear model was found to be 0.06886754

2. LASSO Regression

The LASSO Regression - Lease absolute shrinkage and selection operator is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. I tried to use LASSO as it is generally used when there are a higher number of features, as it automatically does feature selection. [6] I performed a 10-fold crossvalidation, trained the model and tested it on my test data.[7] I then made predictions for the specific 6 months in the following manner - I created a subset of values for October, one similarly for November and December and used these to predict the logerror in those particular time periods. The mean absolute error observed when I used this method was 0.06873633.

3. Ridge Regression

The Ridge Regression is another kind of model used for analyzing multiple regression data that suffer from multicollinearity. [8] When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far fromthe true value. Ridge regression adds a penalty, that shrinks the estimated coefficients towards zero. The mean absolute error observed when I used this method was 0.06879832

4. Random Forest

Random Forest is an ensemble learning method for classification, regression and other tasts that operate by constructing a multitude of decision trees at training time and outputting a class that is the mean prediction of the inidividual trees. Random forests correct for the overfitting of inidivudal decision trees. [9] The training model grows a number of decision trees - you can specify the number, in which at each node only a subset that we specify in the mtry argument are considered for splitting. To train the random forest model to predict logerror, a 5-fold cross validation grid search was carried out on the imputed data set. I used the caret package for this algorithm [10], and chose the method as "Ranger" which is a faster implementation of the Random Forest Algorithm. The best mtry value turned out to be 2; a smaller number is usually needed to keep the trees uncorrelated to each other. The Mean Absolute Error observed here was 0.06812918. The importance of variables was visualized with a variable importance plot.

5. Gradient Boosting Machines

I used XGBoost, an optimized gradient boosting library that has parallel processing, missing values are well-handled internally by this library. The gradient boosting method builds the model in a

[6] Lasso Regression (https://en.wikipedia.org/wiki/Lasso_(statistics))

[7] Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/

[8] Ridge Regression, http://ncss.wpengine.netdna-cdn.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Ridge_Regression.pdf

[9] Random Forests (https://en.wikipedia.org/wiki/Random_forest)

[10] Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem, Luca Scrucca, Yuan Tang, Can Candan and Tyler Hunt. (2017). caret: Classification and Regression Training. R package version 6.0-77. https://CRAN.R-project.org/package=caret

stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. [11] This XGBoost algorithm was a lot faster than all the other algorithms I used. I used the xgboost package in R, did a 10-fold cross validation technique, trained the data and tested it. I followed that by predicting the mean log error at the ten different time points. By using this method, the Mean Absolute Error further reduced to 0.0680411. The variable importance plot observed from the XGBoost algorithm is given in Figure 5.

Summary

Here are the Observed Mean Absolute Log Error's from the models I tried. I am further working on a Ridge Regression Model but do not see it improving the MAE estimates.

ML Technique Used	Mean Absolute Error
Multiple Linear Regression	0.06887
Lasso Regression	0.06874
Ridge Regression	0.06879
Random Forest	0.06813
XGBoost	0.06804

I noticed that the tax features, calculated finished square feet of the house, the year built, the location and count of rooms are the most important features associated with the price estimate of a house. There are also seasonal fluctuations in price, a feature that hasn't been explored in this analysis.

Given the level of missing data, a significant amount of time had to be spent on it, coming up with different imputation strategies to reduce prediction error. I spent a considerable time doing this process, doing multiple iterations as an imputing strategy that I used the first time did not seem to make sense to me when I went back to it. This was an incredible learning experience as I used machine learning alogirthms that I've just read about; there was not enough time to go through the entire gamut of alogirthms. This analysis project helped me figure out a great deal about what I do not still know and understand in the world of Data Science and Machine Learning. An important caveat I faced was lack of knowledge about existing algorithms - my Mean Absolute Error was not close to what Zillow's Zestimate is currently at. I faced issues with not being able to compute/ run a few of the algorithms - among the ones I ran, the gradient boosting model by XGBoost produced the lowest Mean Absolute Error followed by the Random Forest model.

I tried engineering multiple new features based on taxes, room counts etc. but decided against adding them to my final model. A potential weakness was not adding more such engineered features based on important variables - derived factors from already existing variables that might make sense when we look at the problem statement. Similar analyses on the Kaggle learderboard have shown ensemble models with engineered features perform better for the Zestimate challenge than the traditional machine learning algorithms. I wasn't able to get my ensemble model to work properly, the biggest challenge I faced in this project was being unable to debug my errors efficiently as the learning curve was steep.

[11] Gradient Boosting (https://en.wikipedia.org/wiki/Gradient_boosting)

Figures

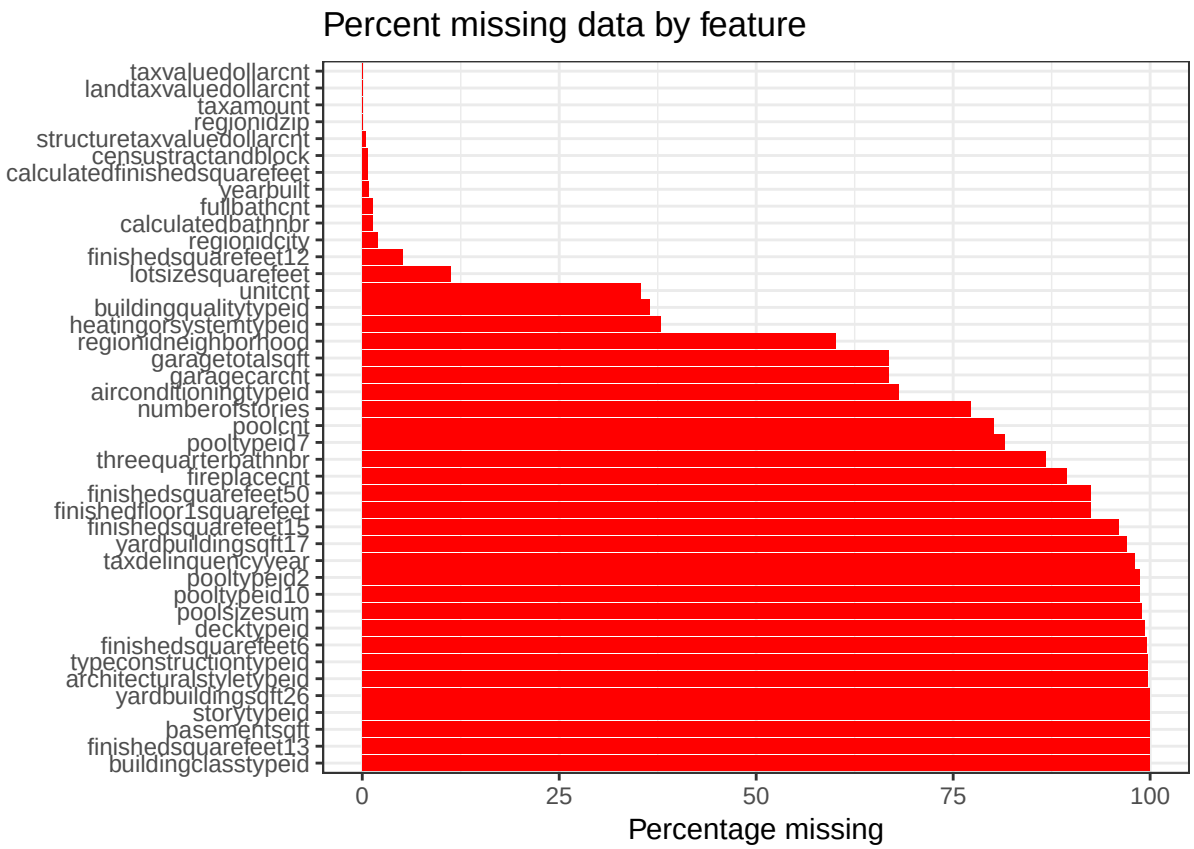

Figure 1. Net Missing data by features in the given dataset

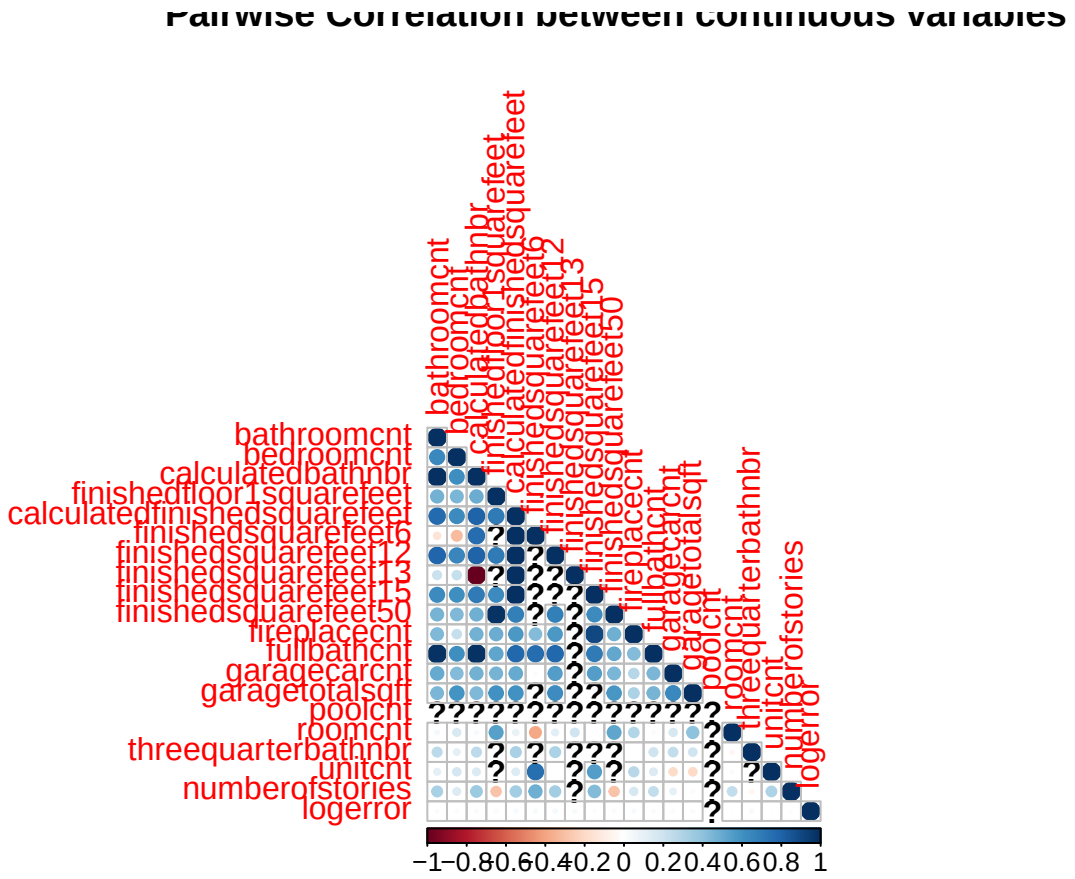

Figure 2. Correlation Plot showing the correlation between continuous features. We see the tax variables being positively correlated.

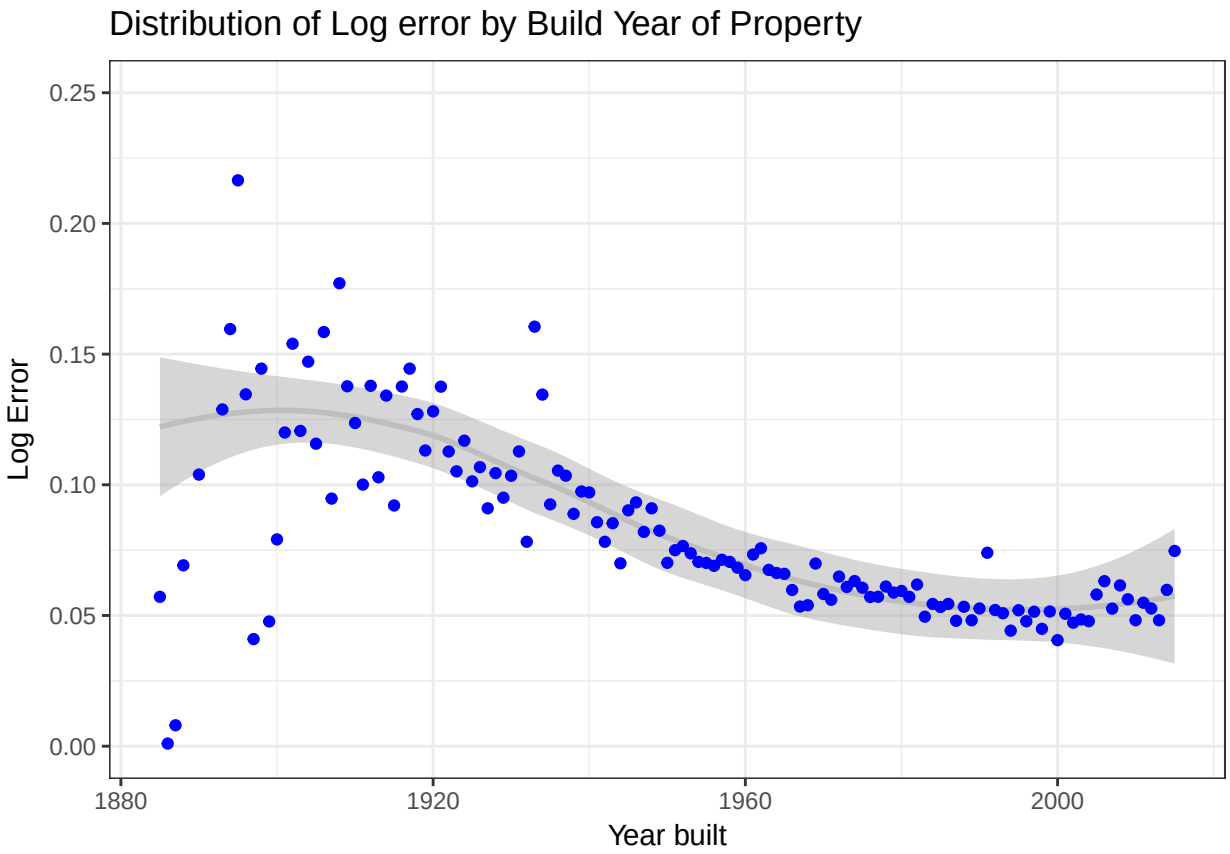

Figure 3. Distribution of Log Error by Build Year

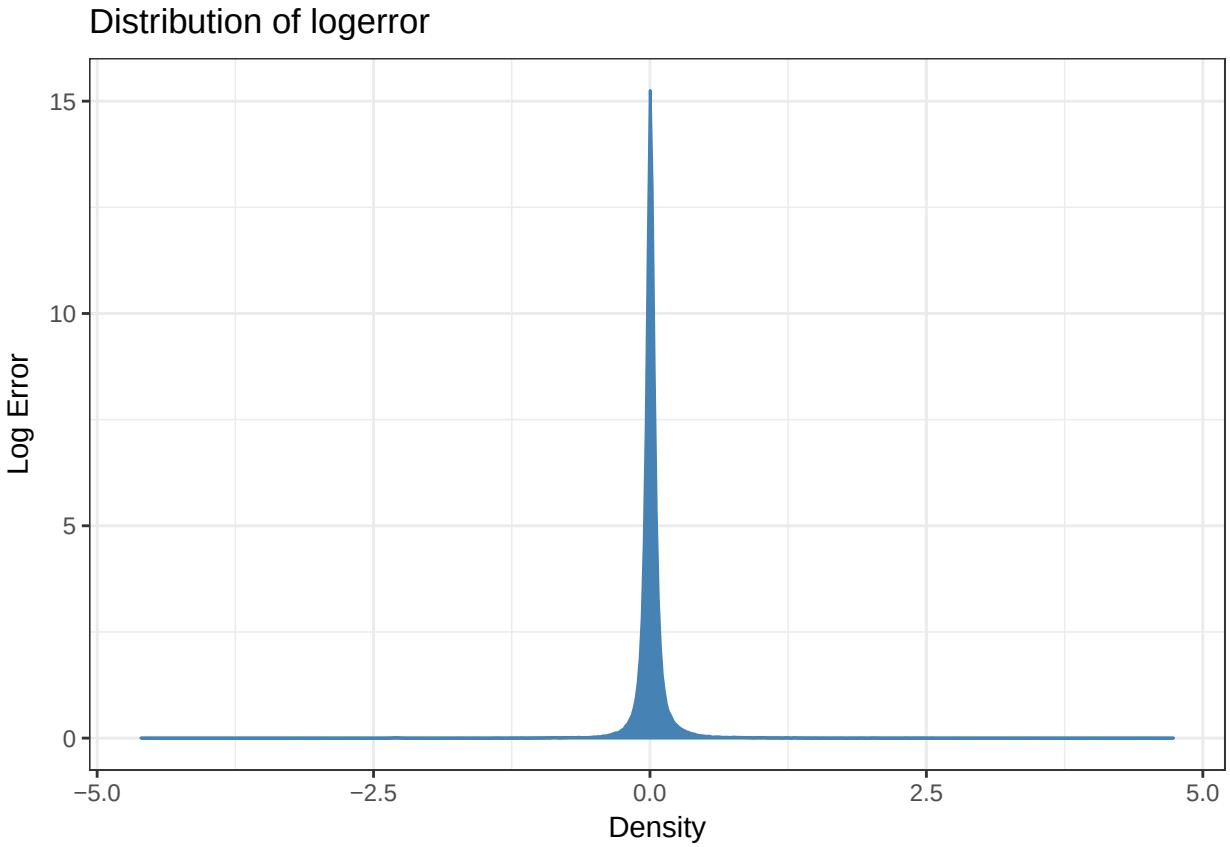

Figure 4. Distribution of the dependent variable, the log error

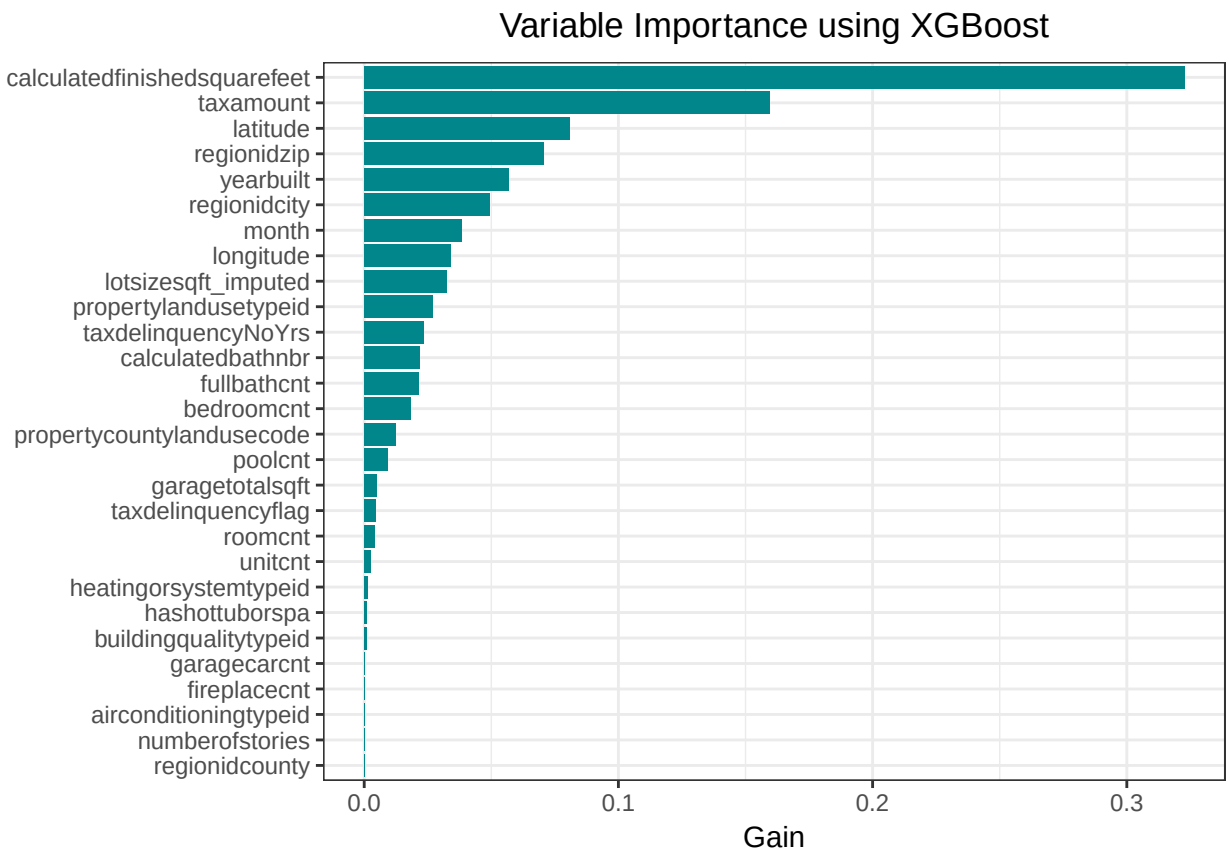

Figure 5. Most important variables according to the XGBoost algorithm