Project Report

Shyamalee

October 11, 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. 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. 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 us to improve the algorithm that changed the world of real estate. 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:

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

This log error is included in the transaction file, and if a transaction did not happen for a property during that period of time, that row is ignored and not counted in the calculation of the Mean Absolute Error. The submission file for this competition should have a log error for every unique property(parcelid) for 6 time points: October 2016, November 2016, December 2016, October 2017, November 2017 and December 2017.

The files provided include a properties file with multiple publicly available and Zillow-owned features for houses in three counties in California - LA, Orange and Ventura respectively. It also contains a train file with parcel id's, transaction dates and log error values and a sample submission file.

Exploratory Data Analysis

The properties dataset was huge, but not all parcelid's in the properties file had transaction information. The properties file was merged with the train dataset (a right join) so that only parcelid's that had gotten sold featured in the data frame used for data analysis. Exploratory analysis was performed by examining all the variables and their distrbutions. 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.

The purpose of the EDA is to figure out how the given dataset looks, come up with visualizations that would help us understand how to proceed with our prediction problem. I did a correlation plot of all the numeric variables - none of the umeric variables were correlated well with the log error. All the tax variables were strongly correlate 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.

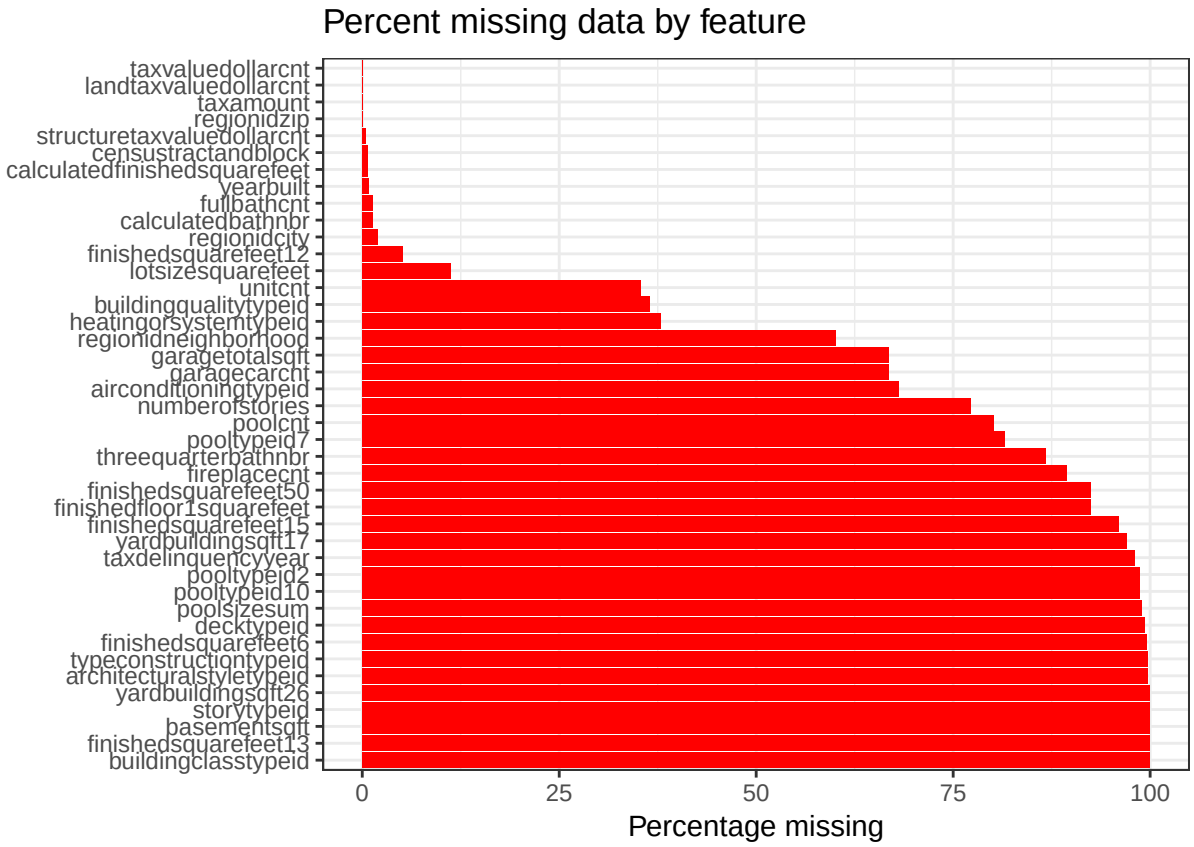

Figure 1 Net Missing data by features in the given dataset

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. When we plot it, I see that it looks normal. I did some more exploration - around how the log error varied over time, the number of transactions over time and leaflet maps of where the properties were located.

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:

- Analyze every variable individually, discard ones with missingness higher than 95 percent
- For highly correlated variables, keep one and discard the other
- 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)
- 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
- Imput 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 and the response variable by fitting a linear equation to the observed data.

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

Logerror was regressed against all of the other variables and the adjusted R^2 value was very low at 0.0053.

2. Elastic Net Regularization - Lasso Technique 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. I used the glmnet package in R for this. I performed a 10-fold crossvalidation, trained the model and tested it on my test data. 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. 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. 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, 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.

4. 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 stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.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. Here is the variable importance plot observed from the XGBoost algorithm.

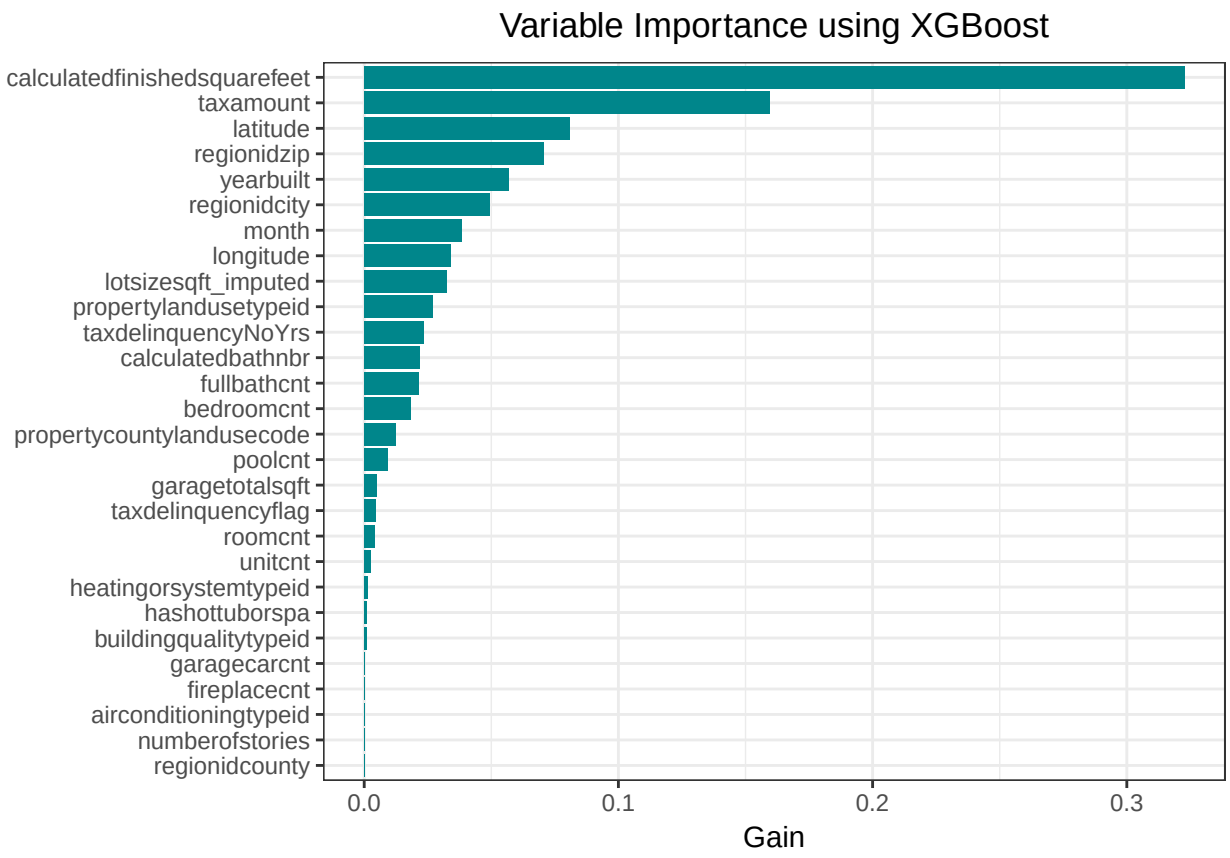

Figure 2. Most important variables according to the XGBoost algorithm

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
Lasso Regression	0.06874
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 even 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 XGBoost model produced the lowest Mean Absolute Error followed by the Random Forest model.

References

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

[2] Wikipedia on Lasso Regression, Random Forests and Gradient Boosting - + Lasso Regression + Random Forests + Gradient Boosting

[3] [Caret Documentation](https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf)

[4] Kernel on EDA in Zillow's Rent Estimate - Kaggle