Archana Ramkumar
Dr. Lucas Mentch
Statistical Learning & Data Science
April 18, 2023

Technical Report

The following analysis conducted on this dataset is motivated by the interests of PA Reality, a new company aiming to gain insights into the property information that influence housing prices in Pittsburgh. My task was to implement a model for house prices, to allow a team of realtors to identify overpriced or underpriced homes by comparing the list price to the predicted house price. The data set contains the unique property identifier, the response variable price, and 16 predictor variables of which five are categorical (description of home, exterior finish of the home, roof material, whether or not there is a basement, and location of zip code in Pittsburgh- either in the city, partly in the city, or not in the city). In general, the 16 predictor variables are typical key property information regarding homes. A few examples are number of rooms, bathrooms, fireplaces, AvgIncome, Distance from downtown and more.

After loading the necessary packages and reading in the test (full_test) and train (full_train) datasets, I first checked for any missing values in any of the columns and found there were no missing values. After examining the type of data for each predictor, I first manually encoded desc, exteriorfinish, Location, and rooftype to convert the categorical variables into a numeric type in preparation to be used in model selection. However, I came to the realization through further research that this would be treating these variables as continuous rather than categorical and would be fully dependent on the arbitrary assignment of the code value for each category. I went back and instead chose the factor approach by converting the data frame to a matrix to introduce one-hot encoding for each of the categorical variables. I made a copy of the original train set (to full_train_t) before doing this so that I could later use that for the tree-based

models. I also made another copy of the original train set (to full_train_b) and used as.factor() to manually convert desc, rooftype, exteriorfinish, and Location to factor variables as I also wanted a dataset without additional columns for the categories. Using the table function, I ended up removing exteriorfinishLog and exteriorfinishConcrete from the full_train set because there were only 1 and 4 observations among 700 train observations respectively. model.matrix() appeared to reduce the number of levels for desc and rooftype.

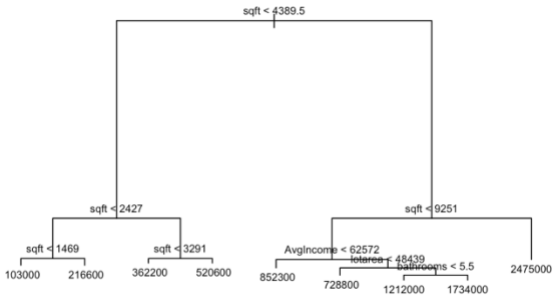
Next, I looked at summary statistics (see Summary Statistics section in R file), various boxplots, and dot plots to visually interpret which predictors seemed to have a strong association with price (see Plots section in R file). Looking at the different plots and correlation matrix (which assume a linear relationship) these were the following correlations of some of the generally more correlated variables with price: sqft (.83), bathrooms (.76), totalrooms (.63), bedrooms (.53), fireplaces (.51). However, there seems to be some collinearity with bedrooms and totalrooms (.84), but totalrooms is more correlated with price. Bathrooms and sqft also have high collinearity (.83), but sqft is more correlated with price. In terms of looking at the summary statistics, I noticed that lotarea had had a min of 0 which led me to consider why that would be case – I would think these are associated with homes that are not multi-family, rowhouses, or singly family homes according to the on-hot encoding of the matrix. Something else to note is the standard deviation of price which was $362.895.80 which tells us a bit about the variability. Looking at the plots, exteriorfinishStone, exteriorfinishStucco, rooftypeSLATE, totalrooms, bedrooms, bathrooms, fireplace, and sqft seemed to have a positive association with price.

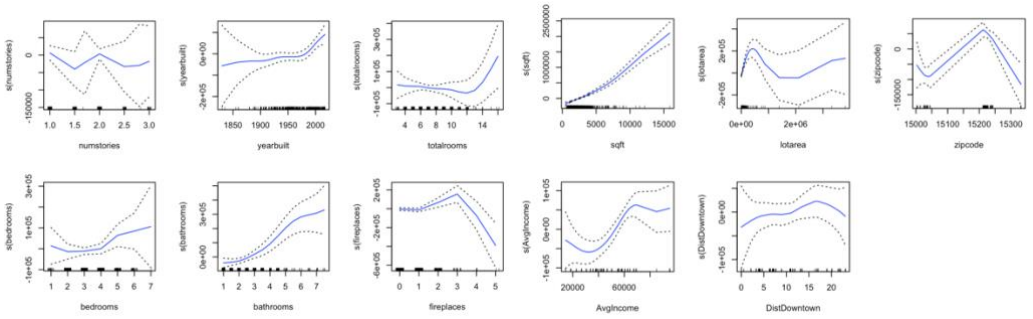
To implement a model that best predicts the price of homes, I considered a variety of models all using 10-fold cross validation. I chose this data split approach over a train-validation split because k-fold gives a better generalizable mean MSE in comparison to a single arbitrary

split of the train set. In total I looked at 10 models (see table below). I started with simple linear regression using forward, backward, and stepwise selection to get an idea of which variables were important in predicting price by looking model selection criterion AIC. With 10-folds, I examined the most frequent variables considered in the final model which had the lowest AIC.

Next, I investigated ridge regression and lasso models to see how the effects of shrinking the magnitude of coefficients by applying a penalty term would affect the overall mean mse. For ridge and lasso, I created the appropriate matrix for the train dataset and to subset out the folds accordingly. Ridge regression resulted in a much larger mean mse than I had expected, and I hypothesize this could be due to biasedness due to systematically under/over-estimate the “true" value of the β's. For lasso, I got a mean mse lower than all three of the linear regression models I performed. Although lasso is similar to ridge, lasso fits a sparse model using only a subset of the predictors and I made sure to use the best lambda value to determine how sparse the model should be (how many coefficients set to 0). Moreover, to examine potential non-linear relationships visually, I chose to explore GAMs to examine individual influence of each predictor on the response due to the additive form (see graphs below). For GAMs, I got a mean mse only slightly larger than the linear regression models by trying subset combinations of the predictors. A much smaller subset of predictors were used compared to ridge regression.

Lastly, I considered tree-based models because they can handle categorical predictors and many predictors well. When building the tree with every predictor in the dataset, the tree only





considered 4 variables in the end: sqft, AvgIncome, bathrooms, and lotarea (see tree above). For regression tree, the unpruned tree with these four variables resulted in the lowest mean mse. For tree ensembles, I used the importance() function to examine which variables seemed most important to find a baseline for model construction. I considered the variables that had the highest %IncMSE and IncNodePurity and generated a couple models for bagging and random forest in a similar fashion. For bagging I considered yearbuilt and numstories in addition to the 4 predictors that were evidently most important (see Bagging section in R file). The test mse was comparable to GAM. For random forests, I considered totalrooms, bedrooms, and yearbuilt in addition to the 4 predictors as mentioned above. For boosting, I first sought to tune the model for shrinkage for learning rate (.09 was the optimal value) and then sought to tune the model for ntree and got 400 (see plots for optimal value under Boosting section of R file). I used the summary() on a boost model fitting all the predictors to determine which predictors had the greatest relative influence to build subsets of boosting models accordingly (see code).

Model	Important Variables	Mean MSE
Linear regression – forward selection	sqft + bathrooms + rooftypeSHINGLE + numstories + exteriorfinishStucco + yearbuilt + `descMULTI-FAMILY` + fireplaces + totalrooms + zipcode + AvgIncome + descROWHOUSE	40315274377
Linear regression – backward selection	`descMULTI-FAMILY` + descROWHOUSE + numstories + yearbuilt + exteriorfinishStucco + rooftypeSHINGLE + totalrooms + bathrooms + fireplaces + sqft + zipcode + AvgIncome	40883352564
Linear regression – stepwise selection	sqft + bathrooms + rooftypeSHINGLE + numstories + exteriorfinishStucco + yearbuilt + `descMULTI-FAMILY` + fireplaces + totalrooms + zipcode + AvgIncome + descROWHOUSE	40343624690
Lasso	**sqft + bathrooms + rooftypeSHINGLE**	**39808435698**
Ridge Regression	"`descMULTI-FAMILY`" + "descROWHOUSE" + "`descSINGLE FAMILY`" + "numstories" + "yearbuilt" + "exteriorfinishFrame" + "exteriorfinishStone" + "exteriorfinishStucco" + "rooftypeROLL" + "rooftypeSHINGLE" + "rooftypeSLATE" + "basement" + "totalrooms" + "bedrooms" + "bathrooms" + "fireplaces" + "sqft" + "lotarea" + "zipcode" + "AvgIncome" + "LocationNotCity" + "LocationPartCity" + "DistDowntown"	86054964302
GAM	yearbuilt + totalrooms + bathrooms + sqft + AvgIncome	41776202543
Regression Trees	(without pruning) sqft + AvgIncome + lotarea + bathrooms	52023484630
Bagging	sqft + AvgIncome + lotarea + bathrooms	41910773783
Random Forests	**sqft + AvgIncome + lotarea + bathrooms**	**35932322124**
Boosting	sqft + bathrooms + lotarea	42691814929

The random forest model with sqft, AvgIncome, bathrooms, and lotarea seemed to have the best predictive performance out of all the models and various predictor subsets that I explored as it had the lowest mean test mse. Looking at the summary table on the previous page, lasso was the second best, and linear regression using forward selection was third best (in terms of predictive accuracy). With random forests, it is more challenging to see how the model treats each of the variables as it is more complex algorithmically compared to linear regression where we can see each of the coefficients for the predictors. So, if the goal is predictive accuracy, random forest would be ideal. If interpretability of what determines house prices is important especially for a team of realtors, linear regression might be easier to understand in terms of the predictor relationships, but there should be some caution with potential nonlinear relationships.

I think the most challenging aspects of this dataset was figuring out how to best work with the categorical variables, especially if there were limited observations of a particular category of a predictor. Converting to a matrix and then back to a data frame allowed me to work with the categorical predictors but the interpretability of one-hot encoding isn't as clear as a continuous predictor. Furthermore, the ambiguity of not knowing what the test prices were also added an additional challenge in comparing to a set baseline. I don't know if I would fully trust my best model in terms of the important variables that the random forest outputted because I'm not sure if my data pre-processing full mitigated all potential issues. I would need to thoroughly reinvestigate the removal of potential predictors, observations, outliers, and transformations again. I hypothesized random forests/boosting to have the highest predictive performance and still think random forest would most likely be one of the best models for this large/complex of a dataset. Something I would like to know in the future is why zipcode was included as a predictor if AvgIncome already gives the average household Income in that zipcode.