ORIE 4740: Statistical Data Mining I
Final Project Report

May 18, 2018

Matthew Kim (yk472)
Alex Wan (fw98)

Abstract

In this paper, we present our study, analysis, and findings of patterns in data regarding homes in Ames, Iowa. Specifically, we use machine learning techniques to discover models that capture meaningful patterns in the data such that these patterns may be used to accurately and consistently predict the sale price of homes in Ames, Iowa. This analysis not only provides an academically compelling and interesting study into the role of home features in the residential real estate market, but also provides practical benefits. Namely, a successful analysis and modeling effort would lead to a "smarter" real estate market in Ames as a whole. Home–buyers would know which features are the most valuable in the market, and thus would be able to make more informed decisions regarding what constitutes a fair price for their choices. Home–owners looking to sell would also benefit, as with this information, they would be able to make smarter and more accurate evaluations of their homes. We first process and clean the data so that missing and unnecessary data and other such inadequacies of the data are dealt with accordingly. Next, we fit several different models that attempt to capture the underlying relationship between the sale price of a home and a myriad of predictor feature variables describing the home, ranging from number of fireplaces to Masonry Veneer type. These models are linear regression with backward stepwise selection, LASSO regression, and a boosting tree model. We compare the models' performance using cross validation and other techniques, and find that boosting trees is the best model. We conclude that this model is the most meaningful and useful in predicting sale price of homes in Ames, Iowa based on home features, and suggest its consideration in future endeavors to predict home sale price.

Contents

1 Introduction

The housing market in the United States has always been and still is, an ever–changing, fluid and more recently quite volatile arena where home–owners and buyers are constantly struggling to sell and buy, respectively, for the best price. With all homes in the United States now having an estimated cumulative worth of $32 trillion, there is no doubt that the housing market is a significant part of the United States' economy and wealth and thus deserves continuous and complex study [3].

Geographic location of homes and the physical land associated with homes have always played a clear and integral role in the determination of home sale prices. Unlike these two general and overarching features however, the relationship between the countless physical features of a home and its sale price is not as obvious, save for very general characteristics such as total square footage and number of bathrooms. A better understanding of this relationship is academically compelling and has the practical benefit of leading to a "smarter" housing market. Therefore, to better understand this relationship, we examine and analyze data regarding homes in Ames, Iowa. The data consists of 1460 records of 79 distinct features of homes in Ames, along with their sale price and ID (i.e. 1460 rows and 81 columns). The source of the data is `kaggle.com`, and it "was compiled by Dean De Cock [of Truman State University] for use in data science education." [1].

In this paper, we fit several different models to meaningfully capture the relationship between home sale price and the myriad of home features provided by the data. The models we fit are linear regression with backward stepwise selection, LASSO regression, and a generalized additive model. We compare these models' performance and conclude that boosted trees is the most superior model.

2 Data Preparation

Before beginning our analyses, we first checked for aspects of the data that needed cleaning or pre–processing. We first removed the `ID` column from our dataset. We then checked if our dataset had any missing values by using the combination of functions `sum(complete.cases(ames))`, where `ames` is the name of our dataset. Fortunately, our dataset had no missing values for every row (the `NA` values are not missing values in the dataset – `NA` is a meaningful level for many of the factor variables that indicate the absence of a particular home feature, **not** the lack of data on that feature).

Next, we checked that all predictor variables of our `ames` dataset were of the correct class. That is, we used the function `sapply(ames, class)` to verify that categorical variables and numerical variables were identified as such. `R` misclassified some of our categorical variables as integer variables – namely the ordinal categorical variables. It also misclassified some integer variables as categorical variables. We used the `as.factor` and `as.integer` functions to correctly reclassify these predictor variables. The variables that were reclassified are: `MSSubClass, LotFrontage, OverallQual, OverallCond, MasVnrArea, GarageYrBlt`.

Finally, we transformed our dataset such that the factor variables were converted to several dummy variables using the `dummy.data.frame` function from the `dummies` package. This resulted in the number of predictor variables increasing to 335, meaning our dataset grew to 1460 rows and 336 columns. This step was useful because it allowed us to drop levels of factor variables, in addition to numeric variables, that exhibited collinearity.

3 Analysis

In this section, we detail our analyses of the dataset, which mainly consist of fitting of several models to the dataset, their evaluation, and their comparison.

3.1 Simple Linear Regression with Stepwise Subset Selection

As a first, exploratory step, we fit a simple linear regression model on the entire dataset. We intentionally began with this naïve approach and fit a simple linear regression model such that `SalePrice` was regressed on all of the other 79 predictor variables because we wished to have a confirmation of the simple hypothesis that there exists **some** linear relationship between at least some of the predictor variables and sale price.

As expected, some of the variables were significant and some were not. However, the linear regression model had a p–value of $p <$ `2.2e-16`, confirming our expectation that the null hypothesis that $\boldsymbol{\beta} = \mathbf{0}$, that is, all coefficients of the predictor variables are zero, may be rejected. In addition, the linear model automatically detected and correctly dealt with collinearity between the variables as expected. For example, `TotalBsmtSF` was automatically excluded from the model, as this predictor is simply the sum of `BsmtFinSF1` and `BsmtFinSF2`. We dropped all of these variables that exhibited collinearity from our model so that future steps we take in analyzing the dataset are not compromised. This shrunk our number of predictors down to 280.

Of course, this simple linear regression model is quite incomplete, as the myriad of predictor variables, many

of which are insignificant, make the model very difficult to interpret. Therefore, we moved on to perform backward stepwise selection on our linear fit. We elected backward stepwise selection, as best subset selection would be intractable because of our large number of predictor variables (best subset selection would require enumerating 2^{335} subset–models).

3.1.1 Backward Stepwise Selection

In performing backward stepwise selection, we first randomly split our dataset as follows, using a set seed for replicability.

Set	Training	Validation	Testing
% of Dataset	50%	30%	20%
Size (# of records)	693	462	305

Table 1: Dataset split

Next, we considered two ways of selecting our final linear regression model with a subset of predictor variables. Namely, we considered selecting our final linear regression model by considering the Bayesian Information Criterion (BIC) v.s. the number of predictors and the cross–validation (CV) mean squared error (MSE) v.s. the number of predictors. For the BIC method, we fit the 280 linear regression models on the training set (1155 records) and selected the model with the lowest BIC on the validation set. For the CV method, we trained the 280 linear regression models on the training set (693 records), and selected the model that returned the lowest MSE when used to predict the sale price of homes in the validation set (462 records).

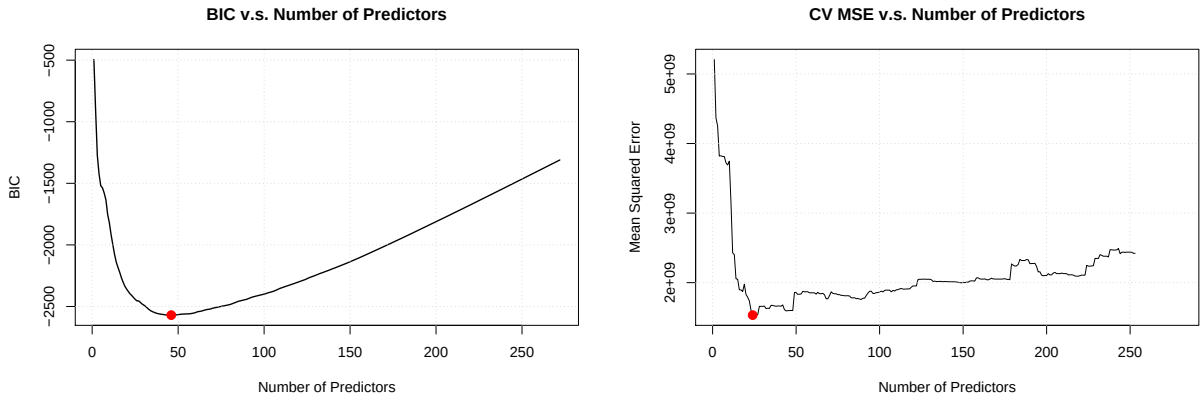

Figure 1: BIC v.s. # Predictors and CV error v.s. # Predictors

The BIC method selected a model with 46 of the 280 predictors, while the CV method selected a model with

24 of the 280 predictors, as shown in figure 1. Finally, we computed the MSE and the mean absolute error (MAE) of the 46–variable BIC model and the 24–variable CV model on the testing set (305 records). The 46–variable BIC model returned the lower error of the two models as shown in table 2, and thus we selected this model as our final linear regression model. Its predictors are shown in figure 3.

Model	46–variable BIC Model	24–variable CV Model
MSE	$1,204,019,219	$2,728,310,818
MAE	$23,476	$27,840

Table 2: Performance comparison between model selected using BIC v.s. CV

```
 [1] "MSSubClass.120"       "MSSubClass.160"       "‘MSZoning.C (all)‘"   "LotArea"
 [5] "LandContour.Bnk"      "LandSlope.Mod"        "Neighborhood.Crawfor" "Neighborhood.NoRidge"
 [9] "Neighborhood.NridgHt" "Neighborhood.StoneBr" "Condition1.Norm"      "Condition2.PosN"
[13] "OverallQual.1"        "OverallQual.2"        "OverallQual.3"        "OverallQual.4"
[17] "OverallQual.5"        "OverallQual.6"        "OverallQual.7"        "OverallQual.8"
[21] "OverallQual.9"        "OverallCond.3"        "OverallCond.4"        "OverallCond.5"
[25] "OverallCond.6"        "YearBuilt"            "YearRemodAdd"         "RoofMatl.ClyTile"
[29] "Exterior1st.CemntBd"  "Exterior2nd.Other"    "BsmtCond.Fa"          "BsmtFinType1.BLQ"
[33] "BsmtFinType2.LwQ"     "Heating.GasW"         "Heating.Grav"         "HalfBath"
[37] "BedroomAbvGr"         "Functional.Maj2"      "Functional.Min1"      "FireplaceQu.NA"
[41] "GarageType.Attchd"    "GarageFinish.Fin"     "GarageCond.Po"        "MiscFeature.Gar2"
[45] "MiscFeature.Othr"     "Exterior1st.CBlock"
```

Figure 2: The 46 predictors of the model selected using BIC

3.1.2 Evaluating the Assumptions of the Linear Regression Model

Finally we check that the assumptions necessary for our final linear regression model hold by examining the plots in figure 2.

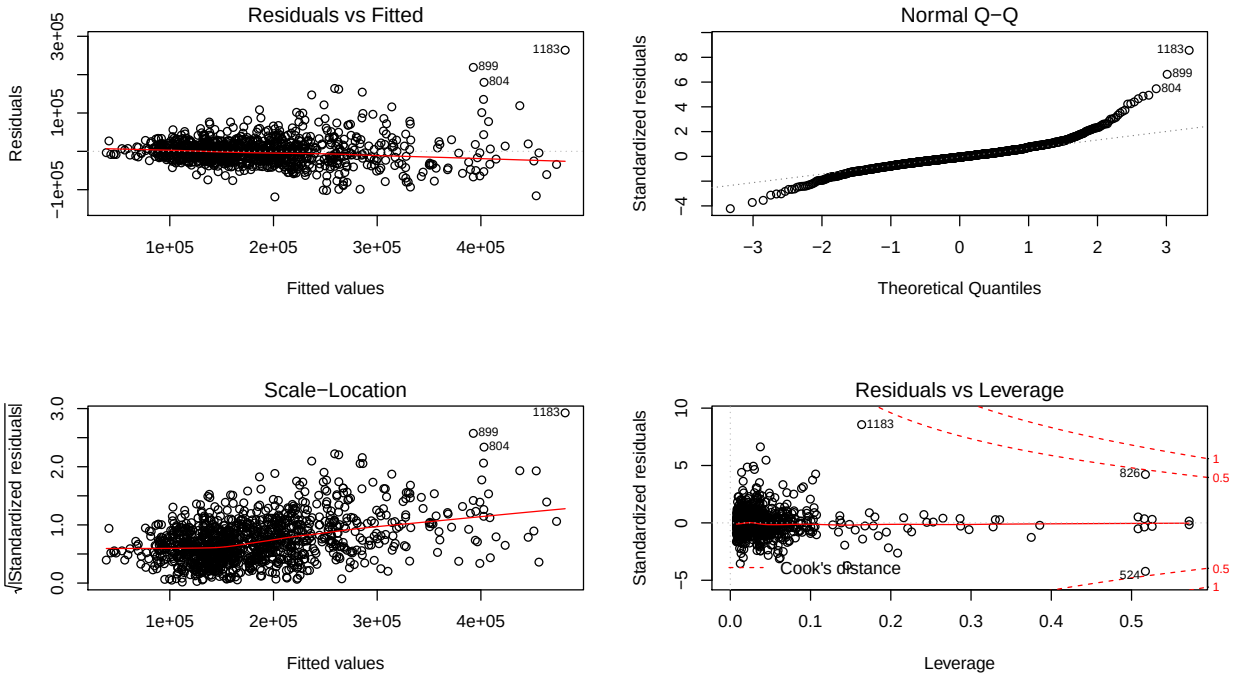

Figure 3: Plots of final linear regression model

We can see that, save for some arguable outliers, the assumptions behind the linear regression are largely held. The Residuals vs Fitted plot does not show a strong nonlinear pattern, the Normal Q-Q plot largely lies on the diagonal with slightly heavy tails, the Scale–Location plot shows that the overwhelming majority of data points lie in a random spread, and finally the Residuals vs Leverage plot shows that there are very few outliers borderline on the Cook's distance lines and the overwhelming majority of points fall well within these lines.

3.2 LASSO Regression

For our second step, we utilized a regularized linear regression model with lasso penalty from the glmnet library. We first constructed a design matrix from the data since glmnet requires the data in this format. The dimensions of the constructed design matrix are 1460 x 295. There are more columns than variables in

the original data since model.matrix() automatically transforms levels in qualitative variables into dummy variables.

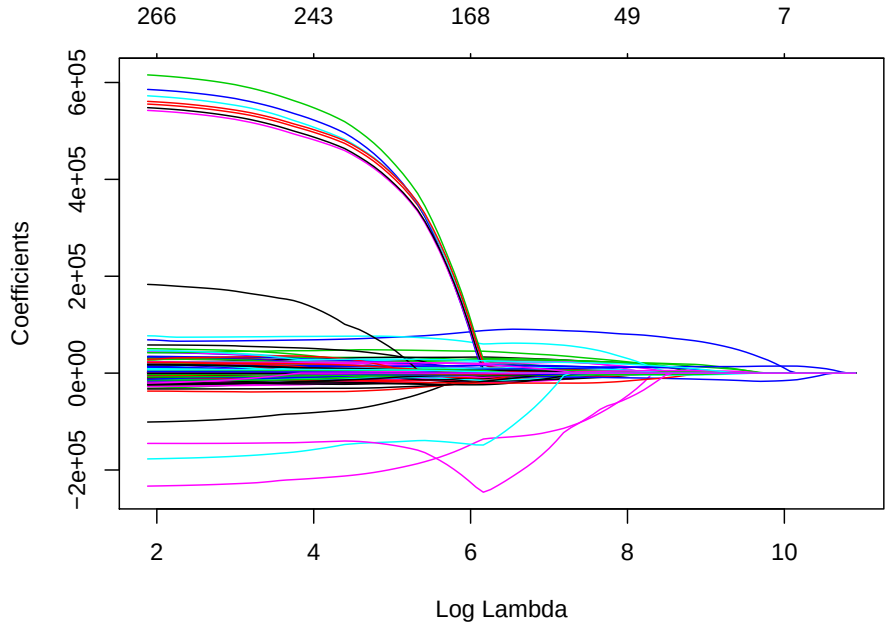

Figure 4: Plots of Coefficients vs. Lambda

Looking at a plot of standardized lasso coefficients as a function of log(lambda), we see that lasso penalty performs variable selection, therefore it can be seen that many of the coefficients are reduced to zero when lambda is large.

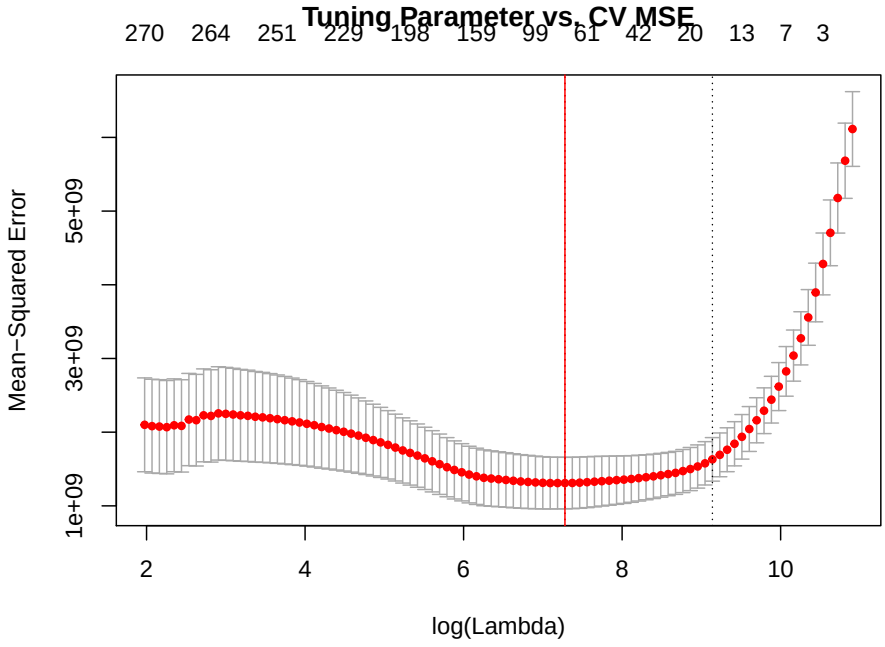

Figure 5: Plots of MSE vs. Choice of Tuning Parameter

In order to find the most optimal tuning parameter, we performed an 80/20 train/test split. cv.glmnet() performed 10-fold cross validation on the training set and provided a lambda that resulted in the lowest cross-validation error, which is marked in red. From our cross-validation, our best tuning parameter is approximately 1449.

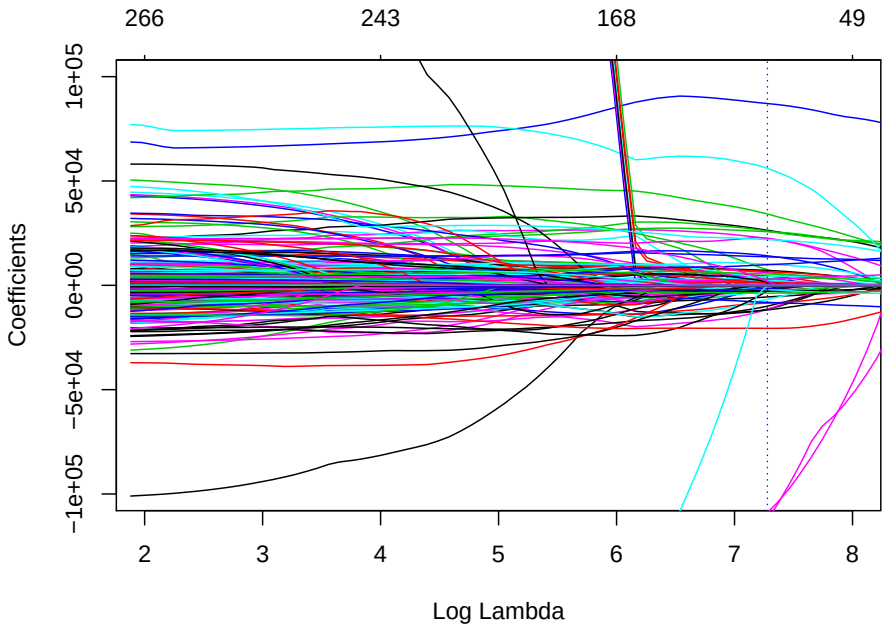

Figure 6: Plots of Coefficients vs. Optimal Lambda

As expected, the lasso with the best lambda yields a sparse model, with many of the coefficients being reduced to zero.

The sparse model trained using the training data contained 78 non-zero coefficients. Thus, we expect that lasso selected significant variables. Although we do not include all 78 variables in this report, many of the variables selected by lasso were also selected by the 46 variable BIC model. These variables include LotArea, specific Neighborhoods,most of the OverallQual and Cond levels, as well as type of Garage and Exterior.

Model	46–variable BIC Model	78–variable LASSO Model
MSE	$1,204,019,219	$1,031,040,572
MAE	$23,476	$18,243

Table 3: Performance comparison between model selected using BIC v.s. LASSO

Finally, we computed the MSE and MAE of the lasso model on the test set and compared it with our previous linear regression model. The lasso regression returned lower errors as shown in table 3.

3.3 Boosted Trees

For our last and final step, we made use of boosted trees from the gbm library.

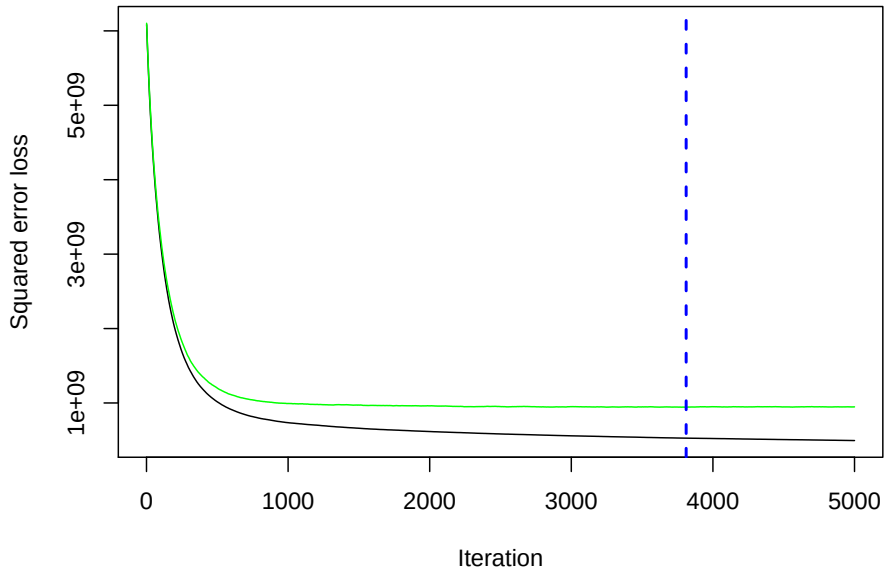

Figure 7: Plots of optimal number of iterations/trees

We used a shrinkage parameter of 0.01 and an interaction depth of 1, as this was stated to be sufficient for most purposes. However, boosting trees tend to overfit as the number of iterations/trees get large, so we proceeded to perform 10-fold cross validation on the same 80 percent training set from before. We found that the most optimal number of iterations is 3811.

	var	rel.inf
OverallQual	OverallQual	29.421124826
GrLivArea	GrLivArea	18.402708605
Neighborhood	Neighborhood	11.721044536
TotalBsmtSF	TotalBsmtSF	6.233592723
BsmtFinSF1	BsmtFinSF1	5.269986473
GarageCars	GarageCars	3.913615050
X1stFlrSF	X1stFlrSF	3.501468106
BsmtQual	BsmtQual	1.864525032
ExterQual	ExterQual	1.832992549
X2ndFlrSF	X2ndFlrSF	1.703763773
LotArea	LotArea	1.660175544
KitchenQual	KitchenQual	1.520132811
FullBath	FullBath	1.304073029
GarageArea	GarageArea	1.191121656
GarageType	GarageType	1.037152796
TotRmsAbvGrd	TotRmsAbvGrd	1.011845521

Figure 8: The most influential variables determined from GBM

Unlike the previous two methods, boosting trees do not perform feature selection. However, we can obtain a table of some of the most influential predictors, which we included above. Once again, we see the same predictors that we selected in previous methods showing up as relatively influential.

Model	LASSO	Gradient Boosted Tree Model
MSE	$1,031,040,572	$879,127,394
MAE	$18,243	$17,626

Table 4: Performance comparison between model selected using LASSO vs. GBM

We once again computed the MSE and MAE for this model on the separate test set, and found that it was even better than the previous lasso model.

4 Conclusion

In our analysis, we found that the boosting tree model had the best performance with both the lowest test MSE and MAE. However, it should be noted that our successive models each had lower test errors than the previous one, it came at the cost of lowering how interpretable it is. Our simple linear regression model is particularly easy to interpret, as it gives insight to the specific variables that increase sale price and by how much. On the other hand, the boosted tree model requires one to input all of the parameters, and it is hard to interpret the difference in sale price due to the change in a single variable.

However, we find across all of our models that several variables show up consistently, whether they were selected through model selection, or based on their relative importance in the case of boosting trees. For example, GrLivArea is significant, since the physical size is one of the most important factors when purchasing a house. Similarly, having a garage is more attractive to buyers in a city where the main mode of transportation is by car. Overall condition and quality as well as the condition and quality of separate areas are significant as well, as expected. Finally, we saw that only some neighborhoods significantly affected the price of the house. This is logical under the assumption of mixed income housing. Overall, our analysis provided an indicator of the most important variables that affect the price of a house.

In conclusion, we find that the boosted tree model is the best model given that we know all of the predictor values, but is less interpretable that the other models. On the other hand, the linear models are more useful when only knowing certain variables and when trying to determine the specific effect of a single variable on the price.

References

[1] De Cock, D. (2016). *House Prices: Advanced Regression Techniques.* [online] Kaggle. Available at: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data [Accessed 18 May 2018].

[2] James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning.* New York: Springer.

[3] Bloomberg (2017). *2017 Was a Robust Year for the U.S. Housing Market, Zillow Says.* [online] Fortune. Available at: http://fortune.com/2017/12/28/house-value-gain-zillow/ [Accessed 18 May 2018].