Predicting House Prices in King County with Machine Learning

Andrew Marion, Matthew Hays, Sanika Kotnis, Michael Black

1. Introduction

Owning a home is a dream or goal that many people hope to achieve. Given the pandemic, many people have wanted to move out of cities and into the suburbs and buy a home. Buying a home can be an incredibly difficult process not knowing if the price is reasonable or if the house will have any problems in the future.

The purpose of this project is to develop and compare models to find the best model to predict the housing price of a house sold in King County in Washington, USA. Having a model to predict the price of a house can be extremely useful for buyers and sellers as each wants to optimize the value of their purchase. For the buyer, knowing the predicted price for a house will allow them to know if they are overpaying or getting a good deal on an undervalued house. For the seller, knowing the predicted price will allow them to not over or undervalued the house. This will ensure they allow them to maximize their profits from selling the house.

1.1 Dataset

The dataset of "House Sales in King County, USA" contains data on houses from King County in Washington, USA sold between May 2014 and May 2015: https://www.kaggle.com/datasets/harlfoxem/housesalesprediction?select=kc_house_data.csv

2. Related Work

There exists quite a lot of related work on predicting housing prices using machine learning techniques. There are examples and write-ups on platforms such as *Kaggle*, *Medium*, *Towards Data Science*, and more. The article that I found the most interesting was from *Towards Data Science*: Predicting House Prices with Machine Learning. The author, John Ade-Ojo, used a Kaggle dataset with home date for 1,460 example houses in Ames, Iowa. Each house has 79 features describing the house. John decided to use Python and Jupyter Notebooks for his analysis.

There were many similarities with John's work and ours. He transformed his data rather than eliminating outliers, as did we. Also like us, John used a regression tree and random forest. Though his model ultimately didn't turn out super successfully (he was only in the top 39% on Kaggle) he ended the article by offering a number of

ways he could improve his results. These included better utilizing categorical variables to reduce bias and hyperparameter tuning. These are things we can keep in mind as well.

3. Methodology

3.1 Data and Exploration, Preprocessing and Cleaning

The dataset contains data on 21,613 house sales from King County between May 2014 and May 2015. It includes 21 fields with some being eliminated as they were not useful in our regression analysis. These included id, date, zipcode, year renovated, and year built.

The dataset did not need to be cleaned as it had 0 null values. The dataset was first explored using functions such as glimpse() and summary() to ensure all data types are numeric. This also allowed for a glimpse at the distribution of the variables. It was noticed that some variables may not have linear distributions and may be skewed. The first problem was price was not linear as shown in **Figure 1**. After attempting different transformations, taking the log of price was the ideal solution as shown in **Figure 2**. It was also determined that sqft_living (square footage of living space), sqft_lot (square footage of the land space), sqft_above (square footage of interior housing above ground level), sqft_basement (square footage of interior housing below ground level), sqft_living15 (square footage of interior housing living space for the nearest 15 neighborhoods), and sqft_loft15 (square footage of the land lots of the nearest 15 neighbors) as each of these variables were found to be skewed.

3.2 Problem Statement

Using the “House Sales in King County, USA” dataset, build and develop models to compare using mean squared error to find the best model.

3.3 Regression Based Models

3.3.1 Simple Linear Regression

We start by fitting a linear regression model on the cleaned dataset. The Adjusted R^2 of this model was 69%, however some variables were not statistically significant because of their high p-values. We then ran the model again after removing the variables that were not significant. However, the Adjusted R^2 was still around 69%/ This suggested the presence of multicollinearity. We then found the multicollinearity index of all the variables used in the model. We found that the multicollinearity index of some of the variables exceeded the cutoff value of 5; so we removed them in the next iteration. However, despite removing those variables and running a model with only significant variables, the Adjusted R^2 was still the same. Thus, simple linear regression didn't prove to be very useful.

3.3.2 Model based on Zip Codes

It is intuitive that different zip codes would have significantly varying starting prices. Therefore we decided to filter out data that only had the same zip code. The goal was to use the zip code with the most number of occurrences since that means more data to work with. We obtained the frequencies of all the unique zip codes in the dataset and found the zip code corresponding to the highest frequency. We then filtered out the data to get a subset with the zip code '98103 (the most commonly occurring zip code)'. We then ran a regression on this and found that the multiple R^2 is 99.3%. This approach showed that a zip code is a very powerful way of predicting the price of a house in the locality.

3.3.3 Forward Stepwise

One type of best subset model selection is called forward stepwise selection. In this, the model selection starts with adding one variable that has the largest reduction in RSS. It will continue to add one variable at a time until it gets to all predictors being added. Next, we used cross validation to compare models.

Forward stepwise resulted in selecting a model with 16 variables due to it having the smallest MSE of 0.0635690033580851. When looking at the simplest model within one standard error, the best model is 12 variables. When comparing 16 variables to 12 variables, both models adjusted R^2 is equal to approximately 0.75, but the 12 variables model has a slightly lower MSE of 0.0640747183735237.

3.3.4 Backward Stepwise

Backward stepwise selection is a similar best subset model selection to forward stepwise selection, but instead of starting with adding one variable at a time, it starts with a model with all variables and drops the least useful one.

Backward Stepwise selection again chose 16 variables as the model with the lowest MSE of 0.0635690033580851. When looking at the simplest model within one standard error, the best model is 13 variables, one more than forward stepwise. When comparing 16 variables to 13 variables, both models adjusted R^2 is equal to approximately 0.75, but the 13 variables model has a slightly lower MSE of 0.0641709329148227.

3.3.5 Ridge Regression

Ridge regression differs from each of the subset selection models by using all variables in the model. It attempts to make the best model by shrinking the coefficients to reduce variance. It uses a lambda as a tuning parameter to reduce the variance. As it increases, the coefficients decrease towards zero and the shrinkage penalty increases.

Ridge regression using all 17 variables resulted in an MSE of 0.0640210770187973 and an MSE of 0.0647437930171121 for the best model within one standard error.

3.3.6 Lasso Regression

Similar to ridge regression, lasso regression is another shrinkage method to making models. The difference between them is that Lasso reduces some variables to 0. By doing this, it is essentially making a best subset model selection when putting variable coefficients to 0. However, it still uses a lambda to shrink the coefficients.

Lasso regression results in a model with 16 variables having the lowest MSE of 0.0653469292488136. The best model within one standard error had 15 variables with a MSE of 0.0653469292488136. These two models were essentially the same.

An interesting note is that even with being a mix between best subset methods and ridge regression, it performed worse than any model so far.

3.3.7 Principal Component Regression (PCR)

Principal component regression, also known as PCR, is a dimension reduction method. PCR is an unsupervised approach to model building where it tries to capture the most variation of the data using linear combinations of variables. The goal is to find the smallest linear combination of variables that account for the variation of the predicted variable.

PCR resulted in the best model being 17 variables based on MSE. However, with PCR wanting to have the smallest number of variables, the 7 variables model was selected as there was diminishing returns after adding the seventh variable. This resulted in an MSE of 0.069442654815093.

3.3.8 Partial Least Squares (PLS)

Partial least squares, or PLS, is another dimension reduction method. The difference between PCR and PLS is that PCR is an unsupervised model method.

The PLS model has diminishing returns after the fourth variable is added. Therefore, the 4 variable model was selected with an MSE of 0.0638798214846813.

There are two interesting notes from this model. First, this model performed better than the PCR model. Second, the model only had 4 variables, far less than all other models so far, and has a very low MSE comparable to Forward Selection and Backward Selection, each with all 17 variables.

3.3.9 Best Subset

R Studio has a *regsubsets* function to determine the best selection of covariates for regression (Lumley, 2020). The maximum number of desired covariates to be considered is specified, and the function returns the list of covariates that minimize the residual sum of squares (RSS) given a certain number of covariates p. For our housing data, a best-subset model was obtained for 1, 2, and all the way up to 17 covariates. The full table listing the best subsets and the 10-fold cross-validated MSE can be found using the link in **Table 1**, and the summaries of the table are written below:

No single covariate was kept in every best-subset selection. The covariates grade, lat, yr_built, and view were chosen very often (16, 16, 14, and 13 times out of 17), while the covariates long, sqft_lot, floors, yr_renovated, and bedrooms were not

chosen very often (1, 2, 5, 5, and 5 times out of 17). The number of bedrooms is a very important predictor in house prices, so why didn't it get picked more? It is important to remember that some covariates are correlated with each other. Bedroom is correlated with sqft_living with a correlation of .58, which makes sense, since additional bathrooms imply more square feet of housing. If sqft_living is already included in the model, then adding bathrooms as a predictor may not provide as much *new* information as adding a different variable, like waterfront for example. The 10-fold cross-validated mean-squared error was smallest for the 16-covariate model (MSE = 0.063569). The 16-covariate model will be a contender for the final model used for the housing data.

Additionally, there is a within-one-standard-error selection method that aims to simplify the model. In the best-subset case, the 12-covariate model has a cross-validated error within 1 standard error of the 16-covariate model. Other models have MSE within one standard error, but the 12-covariate model has the fewest covariates. The purpose of this selection process is to reduce the complexity of the model without compromising too much on predictive power. The cross-validated MSE for the within-one-standard-error model with 12 covariates is MSE = 0.063974.

3.3.10 Likelihood Ratio Test

As it turns out, the models with 11 through 17 covariates are nested; every covariate in the 11-covariate model also appears in the 12, 13, …, and 17-covariate (saturated) model. The same applies to the 12-covariate model and all the way through the saturated model. When regression models are nested like this, the likelihood ratio test (LRT) can be used as a parametric statistical test to compare whether the reduced models are as adequate as the advanced models. The difference in deviances between the two models approximately follows a chi-square distribution, where the degrees of freedom equals the difference in the number of parameters. In the table, model *p* refers to the model with *p* covariates. We will keep the model chosen by the LRT at .05 significance level and compare it to the other models considered in this project.The results of the LRT are displayed in **Table 2.**

As we can see, models 11 and 12 can both be improved upon by adding any number of covariates, so we eliminate them from contention. Models 17, 16, 15, and 14 can be reduced to some degree while still adequately fitting the data. However, no model consistently outperformed every nested model. 14 was sufficient compared to 15, and 13 was sufficient compared to 14. 13 was *NOT* sufficient compared to 15. Model 13, 14, or 15 would be an adequate choice. The model we are choosing based on the LRT approach is model 15. Although 14 vs. 15 was non-significant, as was 13

vs. 14, we are sticking with 15. Model 15 is significantly better than model 13, and when cross-validating mean-squared error (MSE) for models 13, 14, and 15, model 15 had the lowest estimated mean-squared error (MSE=0.06362717). Therefore, the 15-covariate model will also be a contender for the final model used in this project.

3.4 Tree Based Models

3.4.1 Regression Tree

Regression trees use binary recursive partitioning to predict responses. Partitions are made in such a way that they both minimize residual sum of squares (RSS) and leave previous partitions unchanged. The regression tree was made with the *tree* function from R Studio (Ripley, 2021). The first partition was made by splitting the covariate grade at the value of 9, implying that grade is the most important predictor in log house price. The regression tree under default options selected the variables grade, lat, and sqft_living with 8 terminal nodes. The largest partition contains about 16.44% of the data, and the smallest partition contains about 2.89%. The full tree is shown in **Figure 4**.

If an observation had a grade value of 8, a lat value of 46, and a sqft_living value of 8, then the regression tree would predict a log house price of 12.79, which is a price of exp(12.79) = $358,613.33. The *cv.tree* function in R Studio allows cross-validation to be done to estimate MSE. Our estimated regression tree MSE is 0.092610.

3.4.2 Bootstrap Aggregation (Bagging)

One technique developed to improve the regression tree method is bagging. Single regression trees are prone to high variance; if a different training dataset is used, the resulting tree can be significantly different. Bagging, or bootstrap aggregation, uses simulation to achieve a decision tree with lower variance. James et al. (2017) describe the process of bagging in the following way: “... a natural way to reduce the variance and hence increase the prediction accuracy of a statistical learning method is to take many training sets from the population, build a separate prediction model using each training set, and average the resulting predictions.” They note that “averaging a set of observations reduces variance” (p. 316). Each new training dataset obtained with bootstrapping yields a new tree, and the averages are used as the final tree values. As the number of trees increases, the MSE usually decreases, since variance is

decreasing. It is not monotonic decreasing due to the nature of random sampling and bootstrapping. When using 1 through 500 trees, the minimum cross-validated MSE was 0.03264798, which is about a third of the MSE from the single regression tree. Thus, bagging successfully reduced the prediction error and is a better model than the single regression tree.

3.4.3 Random Forest

The random forest method builds upon the bagging process and aims to address a major flaw: highly correlated trees. At each split in the regression tree, random forests randomly sample which covariates are included. If there is one very strong predictor, the bagged trees will almost all split on this predictor, leading to similar-looking trees. James et al. (2017) state, "... averaging many highly correlated quantities does not lead to as large of a reduction in variance as averaging many uncorrelated quantities" (p. 320). The *randomForest* function with default arguments yielded a random forests model for the housing data. **Figure 5** shows the results of the random forest procedure and the variable lat contributes the most to prediction power. Removing lat from the model yields the largest increase in MSE. The random forest model also averages the bootstrap estimates, but with the added benefit of considering different regression trees with each simulation. The cross-validated MSE for the random forest model was 0.03328806, which is a significant improvement compared to the single tree, as expected. However, the prediction error is slightly larger than the cross-validated MSE for bagging.

4. Conclusion

Overall, there were numerous models made comparing the MSE of each model. **Table 3** displays the results of the model selection process. From this, we can conclude that our best model was bootstrap aggregation, or bagging. However, bagging took the most time of any model to run and it used more variables than our smallest model, PLS with only 4 variables. Despite the likelihood ratio model having a statistical test justifying its use, it was still outperformed by other model selection methods. The worst model was the single regression tree. Therefore, if we want the most accurate model, we would not use it.

Appendix

Table 1: Best Subset

https://docs.google.com/spreadsheets/d/1MhQYo5Ehgad94d-bU0E3Sk-9XNSRwvjpw7BEW_hFYro/edit?usp=sharing

Table 2: Likelihood Ratio Test Results

Null Hypothesis: Reduced Model Is Adequate in Fitting the Data
Alternative Hypothesis: Full Model Fits Data Significantly Better than Reduced Model

Symbol	Meaning
*	p-value < .05
**	p-value < .01

Full Model	Reduced Model	Chi-Square Test Statistic	d.F.	*p*-value
Model 17	Model 16	0.01315941	1	0.9087
Model 17	Model 15	0.9416313	2	0.6245
Model 17	Model 14	3.397545	3	0.3343
Model 17	Model 13	6.974427	4	0.1372
Model 17	Model 12	11.20645	5	0.0474*
Model 17	Model 11	16.53694	6	0.0111*
Model 16	Model 15	0.9285	1	0.3353
Model 16	Model 14	3.3844	2	0.1841
Model 16	Model 13	6.9612	3	0.0731
Model 16	Model 12	11.1933	4	0.0245*
Model 16	Model 11	16.5238	5	0.0055**
Model 15	Model 14	2.4559	1	0.1171
Model 15	Model 13	6.0329	2	0.049*
Model 15	Model 12	10.2648	3	0.0164*
Model 15	Model 11	15.5953	4	0.0036**
Model 14	Model 13	3.5769	1	0.0586
Model 14	Model 12	7.8089	2	0.0202*
Model 14	Model 11	13.1394	3	0.0043**
Model 13	Model 12	4.232	1	0.0397*
Model 13	Model 11	9.5625	2	0.002**
Model 12	Model 11	5.3305	1	0.021**

Table 3: MSE Comparison

Model	MSE
Bagging	0.03264798
Random Forest	0.03328806
Forward Selection (Lowest)	0.0635690033580851
Backward Selection (Lowest)	0.0635690033580851
Best Subset Selection (Lowest)	0.0635690033580852
Likelihood Ratio Test	0.06362717
Partial Least Squares	0.0638798214846813
Best Subset Selection (1 S.E.)	0.0639740021498752
Ridge Regression (Lowest)	0.0640210770187973
Forward Stepwise Selection (1 S.E.)	0.0640747183735237
Backward Stepwise Selection (1 S.E.)	0.0641709329148227
Ridge Regression (1 S.E.)	0.0647437930171121
Lasso Regression (Lowest)	0.0653469292488136
Lasso Regression (1 S.E.)	0.0653469292488136
Principal Component Regression	0.069442654815093
Single Regression Tree	0.09032

Figure 1: Normal QQ Plot of Price

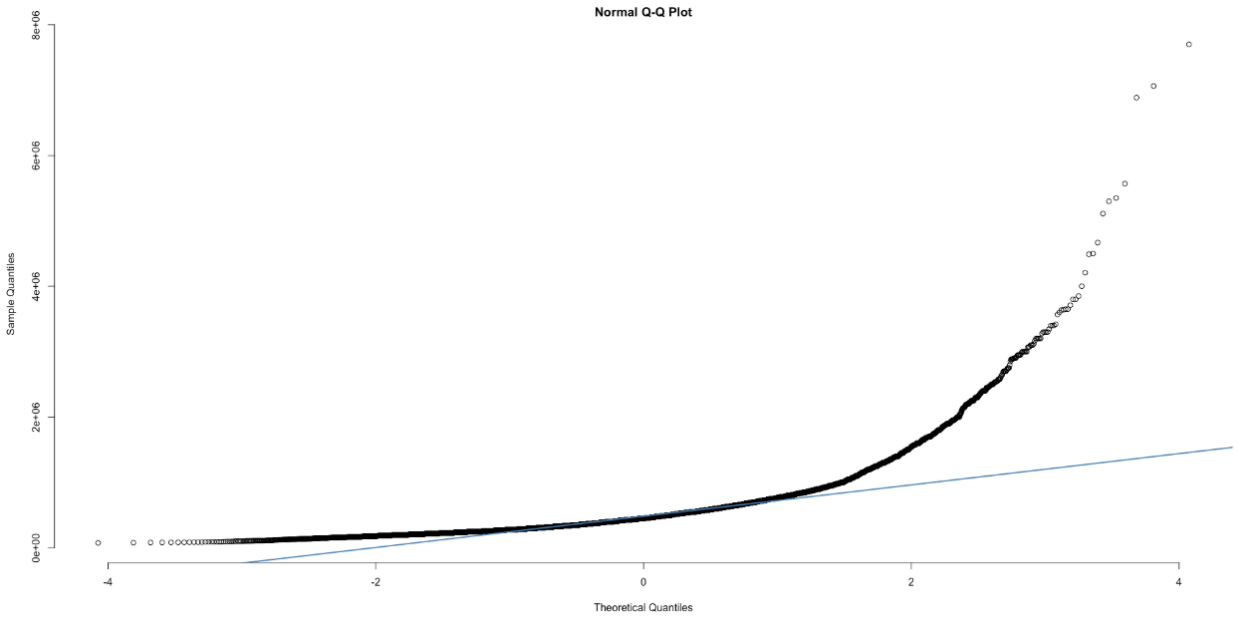

Figure 2: Normal QQ Plot of Price

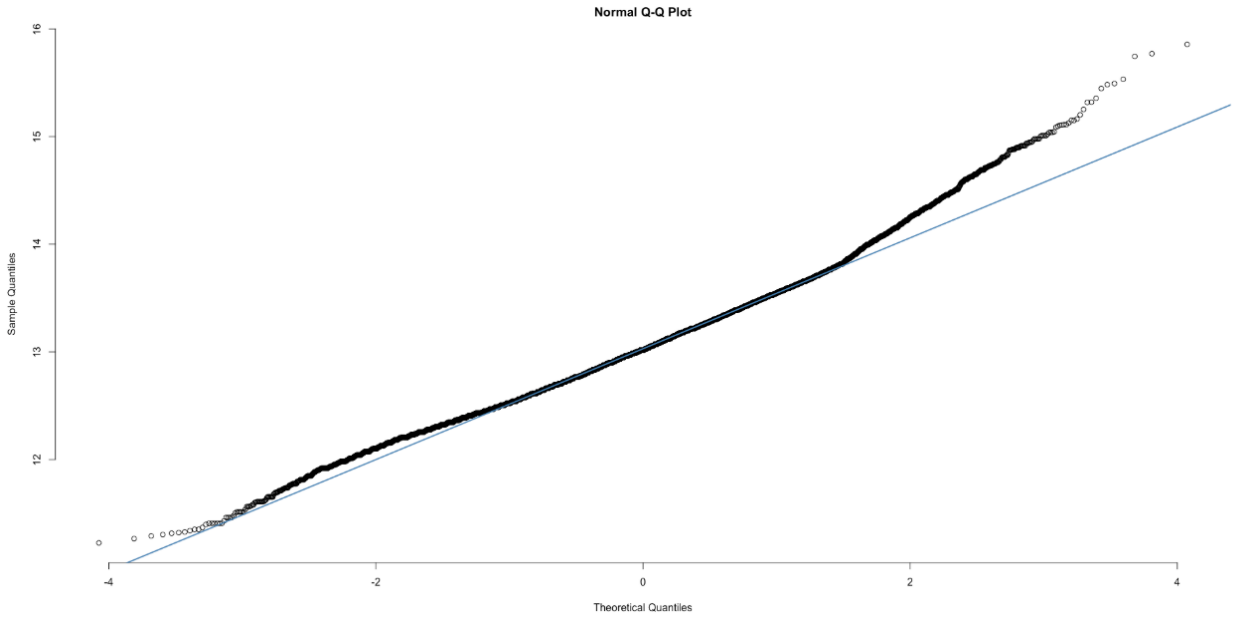

Figure 3: Price Range by ZIP Code

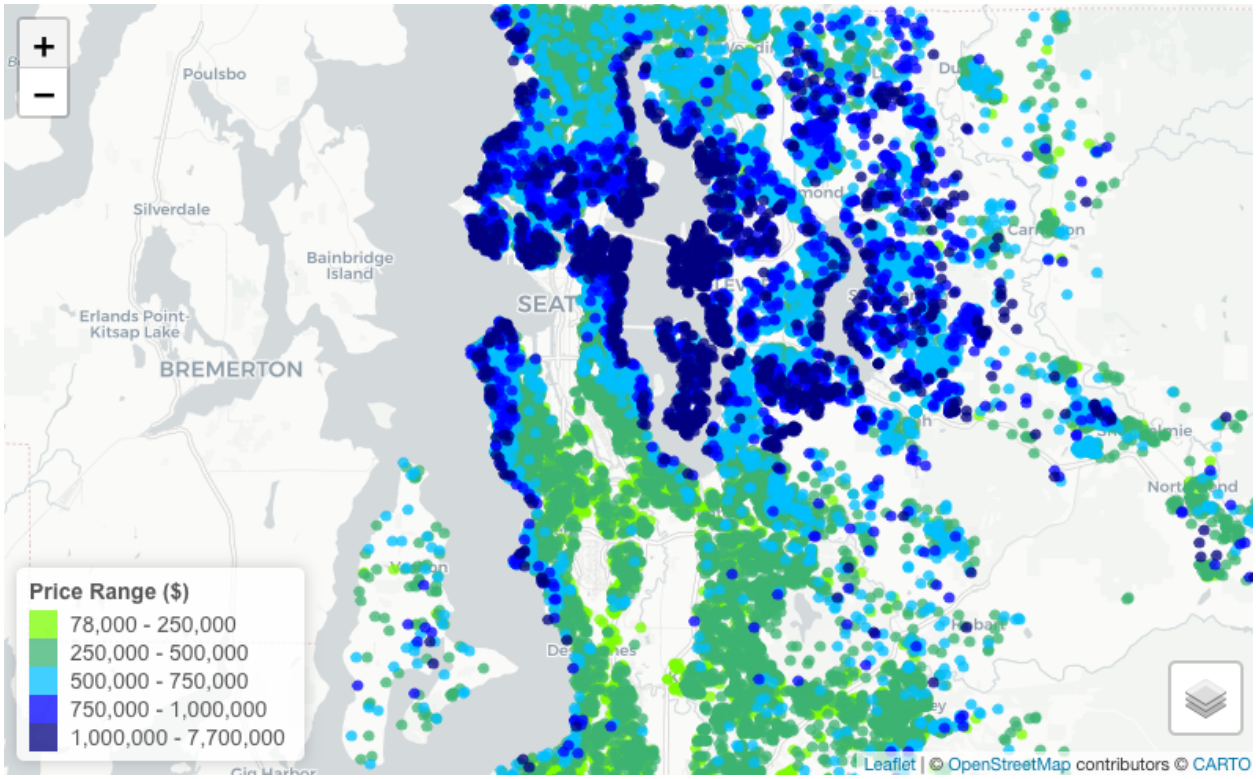

Figure 4: Single Regression Tree

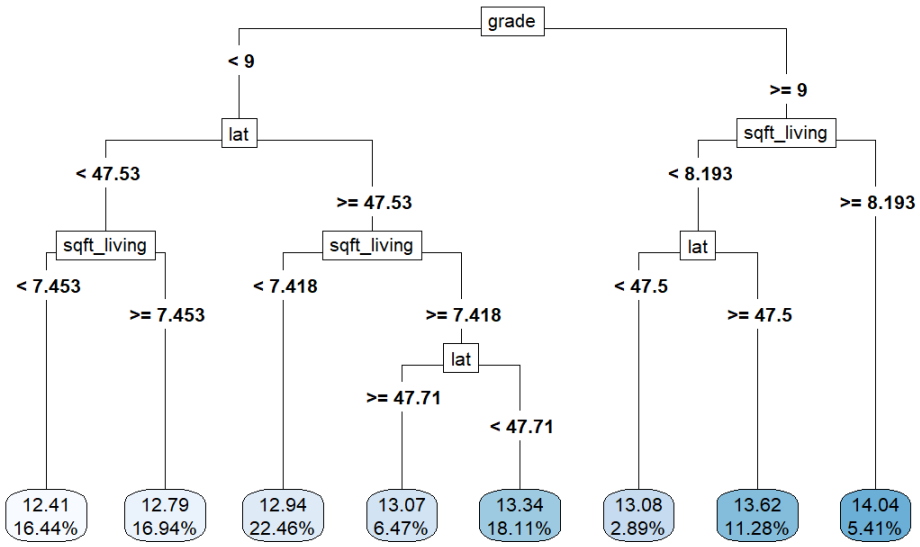

Figure 5: Random Forest Procedure

References

A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18--22.

Angelo Canty and Brian Ripley (2021). boot: Bootstrap R (S-Plus) Functions. R package version 1.3-28.

Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra

Brandon Greenwell, Bradley Boehmke, Jay Cunningham and GBM Developers (2020). gbm: Generalized Boosted Regression Models. R package version 2.1.8. https://CRAN.R-project.org/package=gbm

Brian Ripley (2021). tree: Classification and Regression Trees. R package version 1.0-41. https://CRAN.R-project.org/package=tree

Davison, A. C. & Hinkley, D. V. (1997) Bootstrap Methods and Their Applications. Cambridge University Press, Cambridge. ISBN 0-521-57391-2

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Hadley Wickham (2021). tidyr: Tidy Messy Data. R package version 1.1.3. https://CRAN.R-project.org/package=tidyr

Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2021). dplyr: A Grammar of Data Manipulation. R package version1.0.7. https://CRAN.R-project.org/package=dplyr

Hahn, N. (2020, July 31). Making maps with R. 6 leaflet. Retrieved May 2, 2022, from https://bookdown.org/nicohahn/making_maps_with_r5/docs/leaflet.html

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). An Introduction to Statistical Learning: with Applications in R. Springer Science+Business Media. 10.1007/978-1-4614-7138-7

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 https://www.jstatsoft.org/v33/i01/.

Joe Cheng, Bhaskar Karambelkar and Yihui Xie (2022). leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library. R package version 2.1.1. https://CRAN.R-project.org/package=leaflet

Kristian Hovde Liland, Bjørn-Helge Mevik and Ron Wehrens (2021). pls: Partial Least Squares and Principal Component Regression. R package version 2.8-0. https://CRAN.R-project.org/package=pls

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Stephen Milborrow (2021). rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'. R package version 3.1.0. https://CRAN.R-project.org/package=rpart.plot

Taiyun Wei and Viliam Simko (2021). R package 'corrplot': Visualization of a Correlation Matrix (Version 0.90). Available from https://github.com/taiyun/corrplot

Terry Therneau and Beth Atkinson (2019). rpart: Recursive Partitioning and Regression Trees. R package version 4.1-15. https://CRAN.R-project.org/package=rpart

Thomas Lumley based on Fortran code by Alan Miller (2020). leaps: Regression Subset Selection. R package version 3.1. https://CRAN.R-project.org/package=leaps

Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686