Matthew Horowitz
Chandan Kumar
Dheeraj Reddy Banda

Report

Section 1: Executive Summary

Selling real estate is a major part of the US economy. Determining accurate pricing and knowing whether a home is likely to sell is necessary for understanding the US real estate market. This paper notes that the price and location of a home are the biggest drivers of whether or not a home sells, while price is heavily influenced by location, more than any other feature. In addition, this research evaluated multiple models, and notes that non-parametric tree-based models, plus maps help to best understand the interplay of determining home sales and predicting prices for listing.

Section 2: Overview

2.1 Background on current problem

Residential Real Estate purchases are some of the biggest investments Americans will make in their lives. Sales of homes account for around 15% of US gross domestic product (National Association of Homebuilders, n.d.). As such, for both people working real estate industry, and for those interested in US economic states, determining:

a) The features that determine whether a home sells or not
b) Optimal pricing strategy

A review of academic literature was conducted to determine what past research has found in terms of pricing and predicting sales.

In terms of predicting sales, both Calainho, et al. (2022) and Baldaminos, et al. (2018) noted that discovering the optimal price of a home is key to determine whether or not a sale occurs, with both noting that price is the major factor in determining whether or not a home sells.

In terms of optimal pricing, Forys (2022) notes that prices are a combination of home sizes and features, such as location, taxes, and public valuations.

This research tested these suppositions. In particular, this report examined:

1) Is price the largest driver in determining whether or not a home sells?
2) What features drive price?

2.2 Methodology from Literature Review

In addition to findings on real estate, a literature review was conducted in order to determine the most appropriate methodologies for this research. Particular attention was paid to past methodologies employed, to ensure that model used in this research are appropriate for any type of analysis of the real estate sector.

As a result, past research indicates that most research employ:

1. Parametric models: linear regression for predicting prices, logistic regression for classifying sales or not sales
2. Non-parametric models: such as decision trees—both prediction and classification—ensemble tree methods—like random forests--, black box methods such as perceptron, and geographic information.

Below is a table discussing various model and their uses, from the literature review:

METHODOLGY	**COMMENTS**
Linear Regression	Source: Calainho et al (2022) In the past, this method has been used to predict price of homes, based on attributes of the rental unit. Size, location, type of dwelling—muti-family, condo, single-family
Logistic Regression	Source Wu and Yu (2016) This method has been used to predict sales of homes. Are they likely to sell based on features of a home including price, size location, and location characteristics?
GAM	Source: Grybauskas, et al (2021); Bailey et al (2022) This is often a method used to see if non-linear representations can help be useful in predictions home values.
Decision Tree	Source: Baldominos, et al (2018) Decision trees have been used to examine average prices based on features of a house, as well as chances of selling. This is accomplished by splitting feature space into multiple regions and looking at the resultant
Ensemble Learning	Source: Xiao et al (2022) Ensemble learning is used to add further power to decision trees, through averaging multiple trees: Boosting, Bagging, and Random Forests have all been used.

Section 3: Data

In order to conduct this report, the following data has been selected:

1. USA Real Estate Data Set (Zillow), from Kaggle.
2. Buy vs Rent, (Zillow) from Kaggle.
3. Zip-Code-To-County, From GitHub
4. US Zipcode to County State to FIPS lookup, from Data.world.
5. States.csv file, Created by the Group to help merge data.
6. Fips2County.tsv, from GitHub.

Dataset Description:

1. USA Real Estate Data Set, 512159 observations

Feature	*Data Type*	*Comment*
status	Character	Says whether its ready to be built or has been built and ready to be sold
Bed	Numeric	Number of Bedrooms
Bath	Numeric	Number of Bathrooms
Acre_Lot	Numeric	Size of Lots
City	Character	
State	Character	
Zip_Code	Numeric	
House_Size	Numeric	Square Feet
Prev_sold_date	Date	Date of Last Sale
Price	Numeric	

2. Buy Vs Rent, 31530 Observations

Feature	*Data Type*	*Comment*
Regiontype	Character	Region: country, MSA…
Regionid	Numeric	ID for region
Regionname	Character	
Sizerank	Numeric	Size of region
City	Character	
Countyname	Character	
Metro	Character	
Statename	Character	
Bepropcount	Numeric	
Samplerate	Numeric	
Medbe	Numeric	
Breakeven	Character	Years and Months till home price break even

Medpr	Numeric	

3. Zip-code-to-county, 3236 Oberservations. Used to help merge first two datasets (see data preparation)

Feature	*Data Type*	*Comments*
State	Character	
Statefp	Numeric	State FIP Code
Countyfp	Numeric	County FIP Code
Countyname	Character	
Classfp	Character	Classification

4. US Zip Code to County State to FIPs lookup, 53962 observations, Used to merge the first two datasets (see data preparation section)

Feature	*Data Type*	*Comments*
Zip	Numeric	Zip Code
Stcountyfp	Numeric	State and County FIP
City	Character	
State	Character	
Countyname	Character	
Classfp	Character	

5. States.csv, 50 observations, used to merge datasets 1 and 2 (see data preparation section)

Feature	*Data Type*	*Comments*
State	Character	State Name
Abbreviation	Character	Two Letter Abbreviation for State

6. Fips2County.tsv, 3143 Observations, used to merge Datasets 1 and 2 (see data preparation section)

Feature	*Data Type*	*Comments*
StateFIPS	Numeric	
CountyFIPS_3	Numeric	
CountyName	Character	
StateName	Character	
CountyFIPS	Numeric	
StateAbbr	Character	
State_County	Character	State|County

Section 4: Models Used:

Looking at both the models used in the literature review (see section 2), and the data available (see section 3), the following models were used to answer the questions posed in section 2.

4.1 Prediction of Price:

For predicting prices, the following model was used:

Price predicted based on: bed, bath, acre_lot, house_size, breakeven, new_build, state

Parametric Models:

1. Linear Regression
2. Ridge and Lasso Regression running same structure as linear regression. This was conducted because many of the independent variables were correlated, and variable scaling/elimination was desired.

Non-parametric Models

1. Decision Tree: regression decision tree
2. Random Forrest: regression based random forest

Map of each record based on pricing category, plotted by geo coordinates:

1. Less than 250,000
2. 250,000 to 500,000
3. 500,000 to 750,000
4. 750,000 to 1,000,000
5. 1,000,000 and above

4.2 Classification of Sold and Not-Sold

Sold/Not Sold Classified Based on: price, bed, bath, acre_lot, house_size, breakeven, new build, state

Parametric Models:

1. Logistic Regression

Non-Parametric Models:

1. Decision Tree: Classification
2. Random Forrest: Classification

Map of sold not sold classification, plotted by geo coordinates

Section 5: Data Processing

The following steps were taken to create the necessary data frame needed to run the above models.

5.1 Pipeline Steps

Steps Needed to Join Tables into Single Data Frame

1. Read in the USA Real Estate Data Set as salestotal
2. Check if missing Zip Codes. 197/512159 were missing, and thus records were removed
3. Create a data frame of zip codes
4. Clean up Zip Codes in dataset, by appending leading 0s until there are 5 digits
5. Create dataframe of zip codes
6. Obtain geo coordinates for all properties in dataset, using zipcodeR library
7. Using sqldf merge two dataframes back together, inner join on zipcodes
8. Check all features for missing values, 5762/509115 had missing values, and were dropped
9. Read in the Buy vs Rent dataset as breakeven
10. Read in the Fips2County.tsv file as fips2county
11. Ensure leading 0's are in State FIPS code (2 digits), and County FIPS code(3 digits) exits
12. Merge saleseven and FIPS codes with leading 0s using sqldf, inner join on state abbreviations and county names.
13. Remove the duplicate county name
14. Read in US Zip Code to County State to FIPs as zipcode2fips
15. Select out the columns of: ZIP, STCOUNTYFP, COUNTYNAME, using sqldf
16. Add leading 0s to all FIPS codes in this new dataframe
17. Read in ZIP-COUNTY-FIPS_2018-03.csv as zipcode2fips
18. Extract ZIP, STCOUNTYFP, COUNTYNAME, STATE from zipcode2fips using sqldf, as zip2fips
19. Ensure leading 0s as on all zipcodes have leading zeros in the zip2fips data frame
20. Merge the breakeven data with zip2fips using sqldf, inner join on county fips codes, new data frame named even4
21. All duplicate columns removed from even4
22. Breakeven, zipcode, and fips were extracted from even4, and then cbind into dataframe even5
23. Next, using library substringr, the string for breakeven, kept as string with words year and month(s) included, were transformed into numeric values representing months
24. Year values were extracted, converted to numeric and multiplied by 12
25. Next, the month number was extracted, converted to numeric. This was then summed to the year values (expressed in month form).
26. This gave us a breakeven value (in months per zip code) and placed into data frame called even6
27. However, even6, as a result of inner joins on FIPS codes resulted in multiple breakeven dates for each zipcode. As a result, the median value per zipcode was taken and placed into mediansbreakeven, ensuring one break even value, with minimum bias, (number of months) per zipcode.
28. Using sqldf salestotal and mediansbreakeven were merged on inner join on zipcodes into data frame named saleseven
29. Columns were renamed to help in analysis

5.2 Initial Cleaning: Pre-Summary Statistics

Steps:

1. Check to see if any features in saleseven were missing data.
    a. Bed had 95758 missing values
    b. Housesize had 112989 missing values
    c. Bath had 92129 missing values

   d. Acre Lot had 103546 missing values
2. Due to high number of missing values, the following records could not be deleted. Due to the heterogeneous nature of homes, the best value to put into these missing values was median value, as it an average value that is not affected by extremes, unlike mean, for house size, bedroom, bathroom, acre lot respectively.
   a. Median house size was 1744 square feet
   b. Median bedroom is 3
   c. Median bathroom 2
   d. Median Acre Lot is .5

   These values were entered into the observations with missing values.

3. Next, multiple repeated columns were removed.
4. In order to determine if a house was sold, records without dates were coded with a 1, all others 2. This was cbind into the saleseven data frame and then renamed has_sold.
5. Bed, bath, were set to factor using as.factor
6. Status, was encoded with 1 meaning it is built and existing, ready for sale, while 2 is ready to build
7. This was cbind into the saleseven and renamed new_build
8. Redundant data was removed by using select distinct * in sqldf
9. Finally, while not suitable for merging, Zip-code-to-county, was used to cross validate the data quality of the final data frame.

The resulting data frame, with 503,240:

Features	Data Type	*Comments*
Status	Character	For sale or ready to build, dummy coded in the new_build
Price	Numeric	
Bed	Numeric	
Bath	Numeric	
Acre_Lot		
Full Address	Character	Address including number, street, apartment (if applicable), city, state, zipcode
Street	Character	Number plus street
City	Character	
State	Character	
House_size	Numeric	
Sold_Date	Numeric	Date of sale if applicable, dummy coded in has_sold
Zipcode	Character	Character version needed for leading zeros
Lat	Numeric	Latitude
Lng	Numeric	Longitude

Breakeven	Numeric	Months to break even
Has_sold	Character	Dummy Coded 0 for not new build 1 for new build
New_Build	Character	Dummy Coded 0 for not new build 1 for new build

Section 6: Data Analysis

6.1 Summary Statistics

Five Summary Statistics on Continuous Variables/ Min/Max Count Data for Discrete Variables (only variables included in modeling), including price and has_sold the target variable for both prediction and classification respectively:

1. Calculated by calling Summary() function on saleseven
2. Boxplots were run on all continuous variables to visualize any outliers or leverage points
3. Histograms were run on all discrete variables

During this process, it became apparent that most records were repeated in original USA Real Estate Data Set.

Below are the outputs of the original summary statistics, description of issues, removal of replicated data, and variable transformation:

Pre-Transformation (Note, dummy variables in character have been converted to numeric):

Summary Statistics:

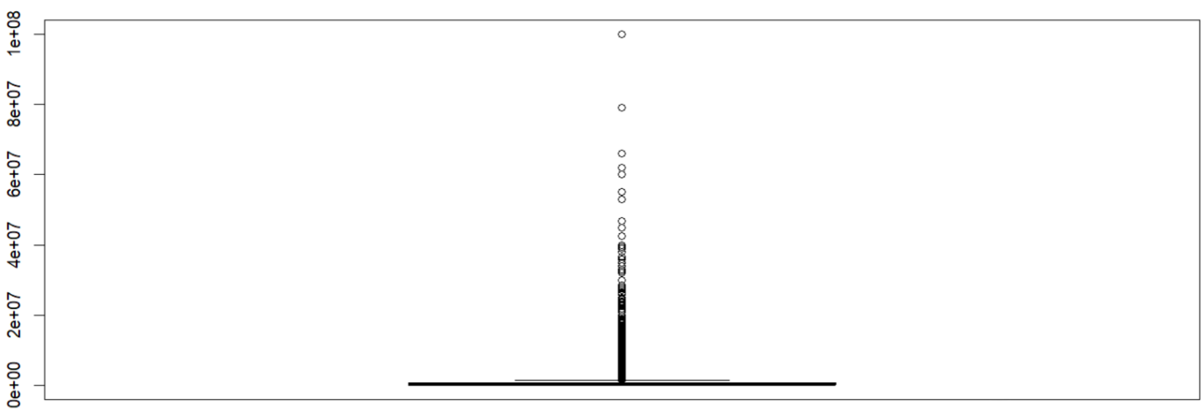
```
   status              price                bed              bath             a
cre_lot         full_address
 Length:503240      Min.   :        0   3      :238010   2      :246999   Min
.   :      0.00    Length:503240
 Class :character   1st Qu.:   224900   2      : 87458   3      :102168   1st
Qu.:      0.23   Class :character
 Mode  :character   Median :   399900   4      : 81651   1      : 87762   Med
ian :      0.50    Mode  :character
                    Mean   :   709112   5      : 28260   4      : 36509   Mea
n   :     15.47
                    3rd Qu.:   725000   1      : 27125   5      : 13957   3rd
Qu.:      1.30
                    Max.   :100000000   6      : 19038   6      :  7002   Max
.   :100000.00
                                        (Other): 21698   (Other):  8843
    street              city               state             house_size
sold_date          zipcode
 Length:503240      Length:503240      Length:503240      Min.   :    100   L
ength:503240      Length:503240
 Class :character   Class :character   Class :character   1st Qu.:   1342   C
lass :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Median :   1744   M
ode  :character   Mode  :character
                                                          Mean   :   2107
                                                          3rd Qu.:   2300
                                                          Max.   :1450112

      lat              lng            breakeven         has_sold        new_build
```

```
 Min.   :17.98   Min.   :-81.50   Min.   : 1.00   Min.   :1.000   Min.   :1.0
00
 1st Qu.:41.66   1st Qu.:-72.80   1st Qu.:20.00   1st Qu.:1.000   1st Qu.:1.0
00
 Median :42.28   Median :-71.83   Median :28.50   Median :1.000   Median :1.0
00
 Mean   :41.46   Mean   :-71.76   Mean   :35.75   Mean   :1.402   Mean   :1.0
02
 3rd Qu.:42.89   3rd Qu.:-71.08   3rd Qu.:42.00   3rd Qu.:2.000   3rd Qu.:1.0
00
 Max.   :47.32   Max.   :-65.28   Max.   :87.50   Max.   :2.000   Max.   :2.0
00
```

The above data was used to a) check if min and max values fell in allowable values, which all features did. Also, continuous features' median's and interquartile ranges were checked to see if extreme values were present. Bathroom, Price, Acre_Lot all had extreme values on the positive side. Thus, box-plots for continuous data and histograms for discrete data were run to better visualize distribution patterns.

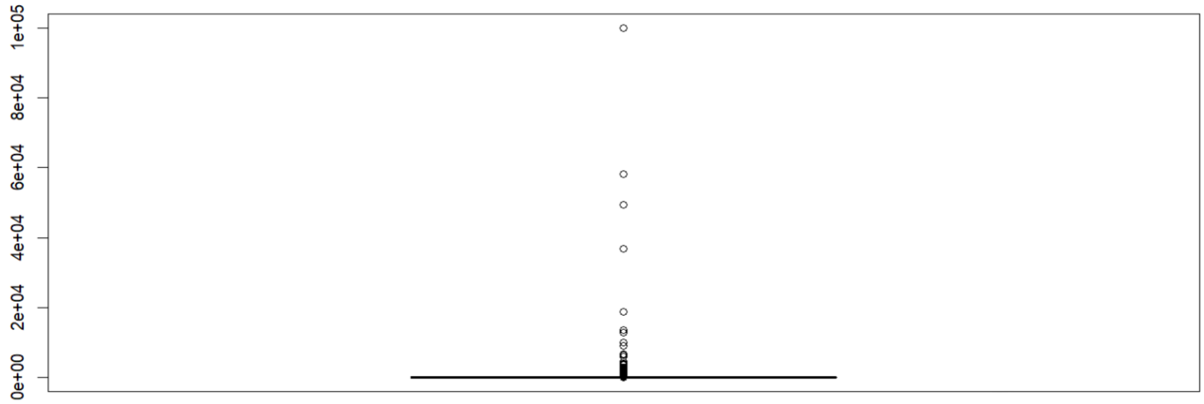
<u>Box Plots for Continuous Data</u>

a) saleseven$price



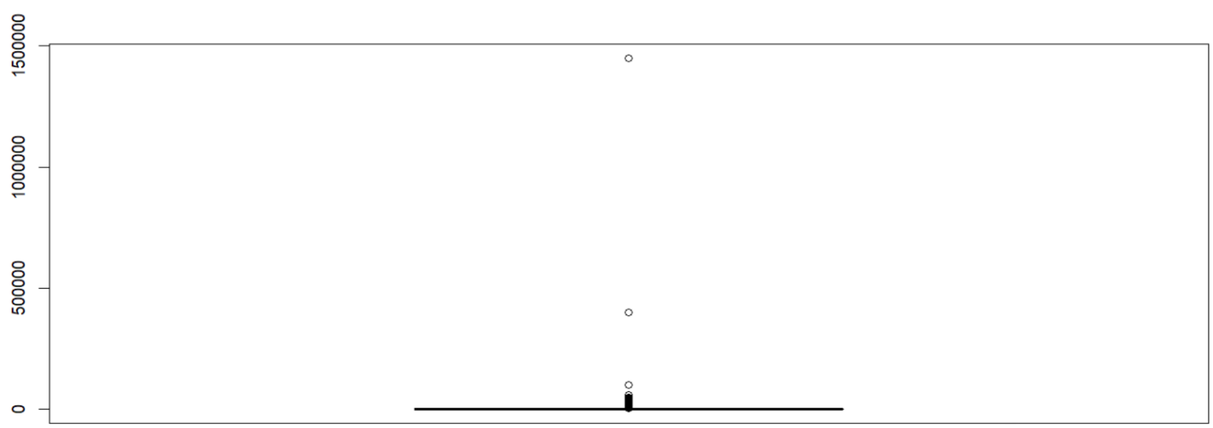
Here, we can see a positive skew in the price data, which was fixed in the transformation step.

b) saleseven$acre_lot



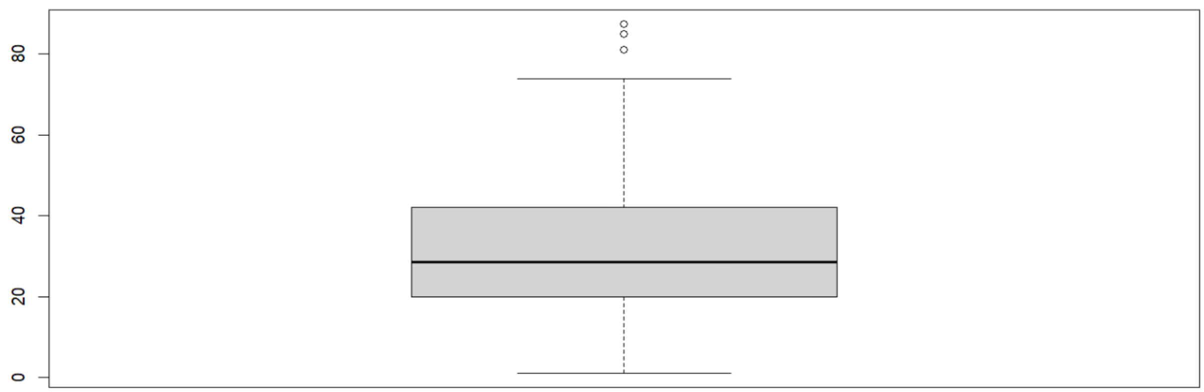
Similarly, acre_lot is extremely positively skewed, and was transformed, see next section.

c) saleseven$house_size



House size also showed high levels of positive skew, and was transformed, see next section.

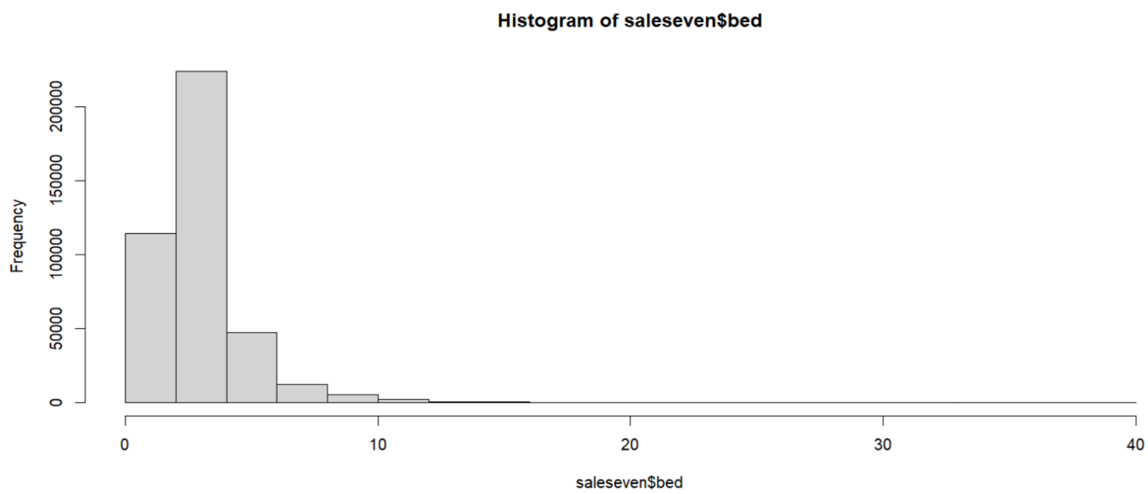
d) saleseven$breakeven



Break even exhibited distribution with limited skew, no changes were taken.

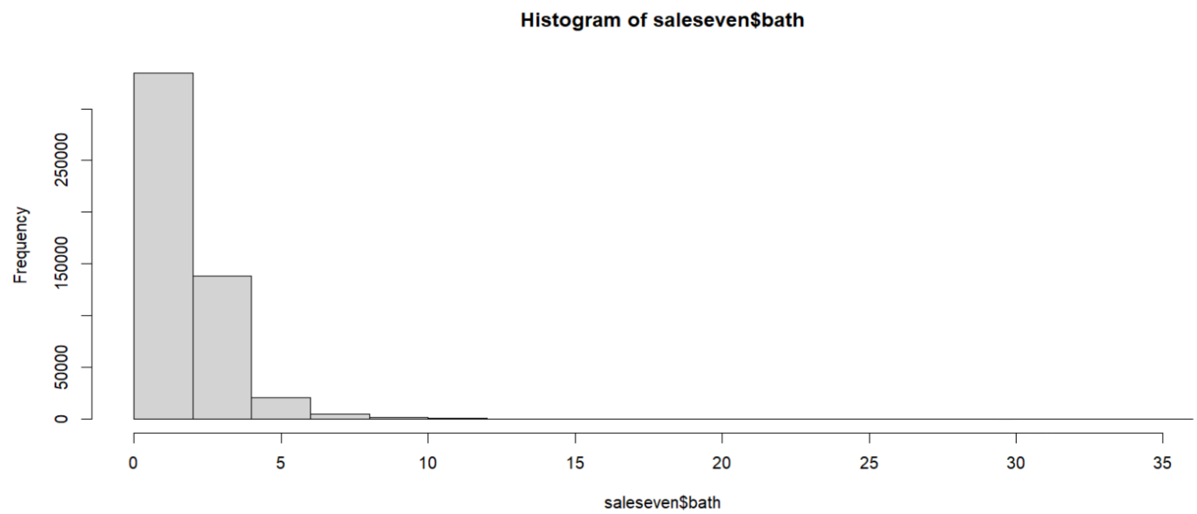
Histograms for Discrete Data

a) saleseven$bed



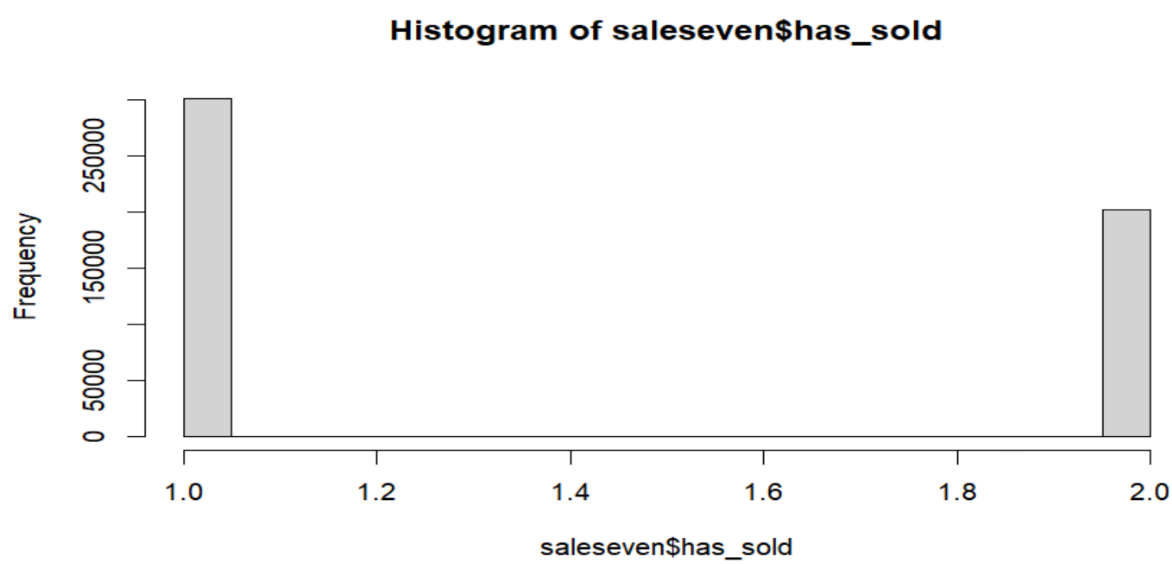
There was a strong positive skew, and as such beds was transformed

b) saleseven$bath



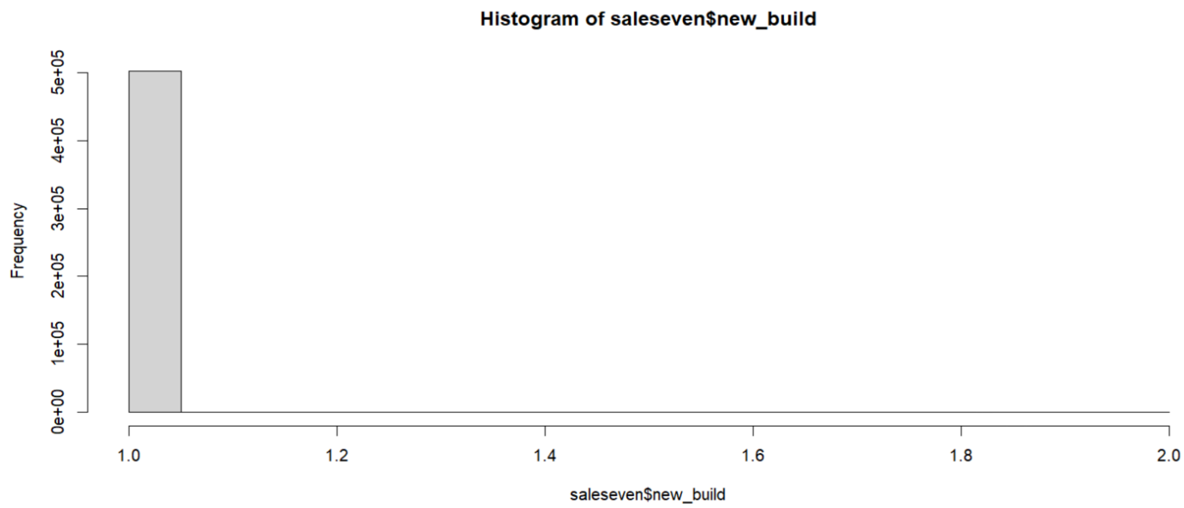
Bath has had a high level of positive skew and was transformed.

c) saleseven$has_sold



Has sold showed strong balance, no changes needed.

d) saleseven$new_build



Even though data was very unbalanced, no transformations were made.

6.2 Removing Duplicates:

Upon further inspection of the data, duplicate values were found. Once duplicates were removed, 63861 observations remained. Below are summary statistics, box plots, and histograms after removing duplicates.

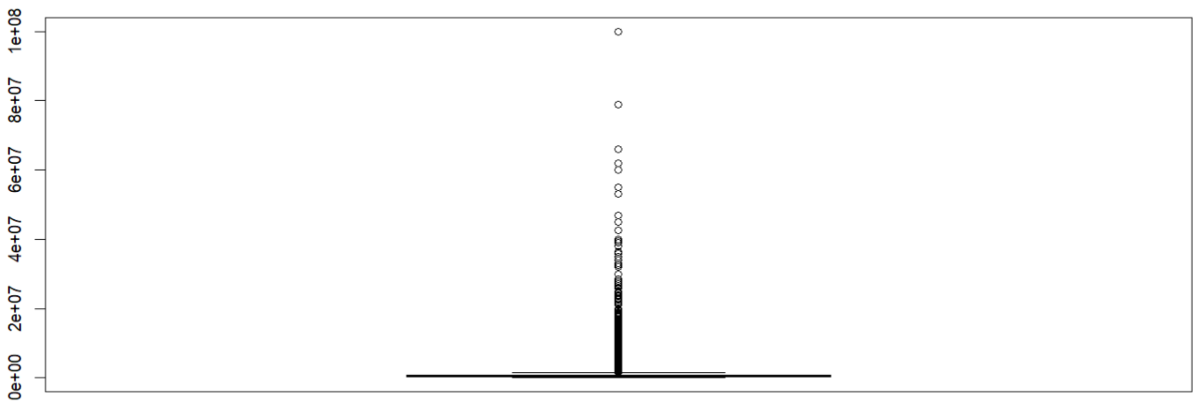
Summary Statistics:

```
status                price                 bed               bath             acre_lot
full_address
 Length:63861         Min.   :        0   Min.   : 1.000   Min.   : 1.000   Min.   :
0.00   Length:63861
 Class :character     1st Qu.:   250000   1st Qu.: 3.000   1st Qu.: 2.000   1st Qu.:
0.20   Class :character
 Mode  :character     Median :   450000   Median : 3.000   Median : 2.000   Median :
0.50   Mode  :character
                      Mean   :   862858   Mean   : 3.322   Mean   : 2.471   Mean   :
17.01
                      3rd Qu.:   799900   3rd Qu.: 4.000   3rd Qu.: 3.000   3rd Qu.:
1.00
                      Max.   :100000000   Max.   :40.000   Max.   :36.000   Max.   :1000
00.00
    street              city               state              house_size        sold_date
zipcode
 Length:63861         Length:63861       Length:63861       Min.   :    100   Length:638
61       Length:63861
 Class :character     Class :character   Class :character   1st Qu.:   1401   Class :cha
racter   Class :character
 Mode  :character     Mode  :character   Mode  :character   Median :   1744   Mode  :cha
racter   Mode  :character
                                                            Mean   :   2097
                                                            3rd Qu.:   2108
                                                            Max.   :1450112
      lat              lng           breakeven        has_sold       new_build
```

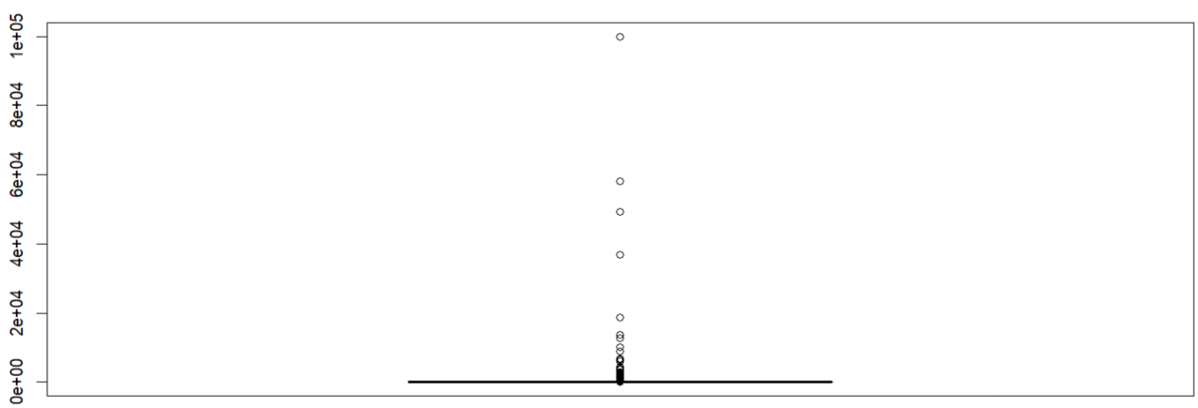
```
 Min.   :17.98   Min.   :-81.50   Min.   : 1.00   Min.   :1.000   Min.   :1.000  
 1st Qu.:40.76   1st Qu.:-73.97   1st Qu.:16.50   1st Qu.:1.000   1st Qu.:1.000  
 Median :41.51   Median :-72.91   Median :25.00   Median :1.000   Median :1.000  
 Mean   :41.05   Mean   :-72.40   Mean   :29.35   Mean   :1.457   Mean   :1.001  
 3rd Qu.:42.35   3rd Qu.:-71.36   3rd Qu.:37.00   3rd Qu.:2.000   3rd Qu.:1.000  
 Max.   :47.32   Max.   :-65.28   Max.   :87.50   Max.   :2.000   Max.   :2.000  
```

Boxplots:

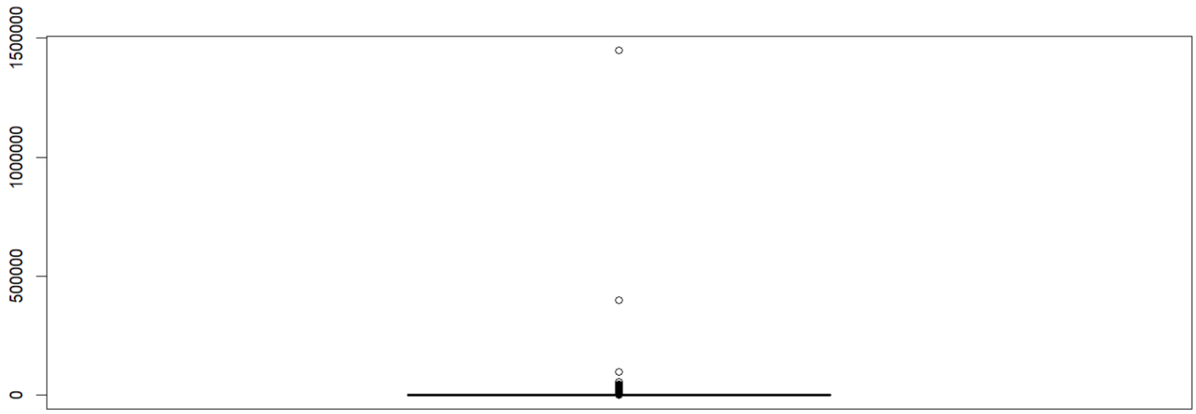
a) saleseven$price



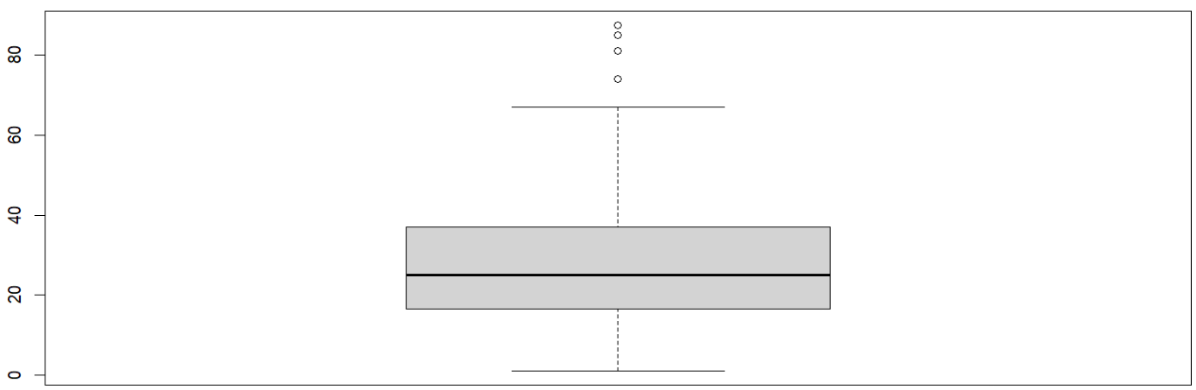
b) saleseven$acre_lot



c) saleseven$house_size

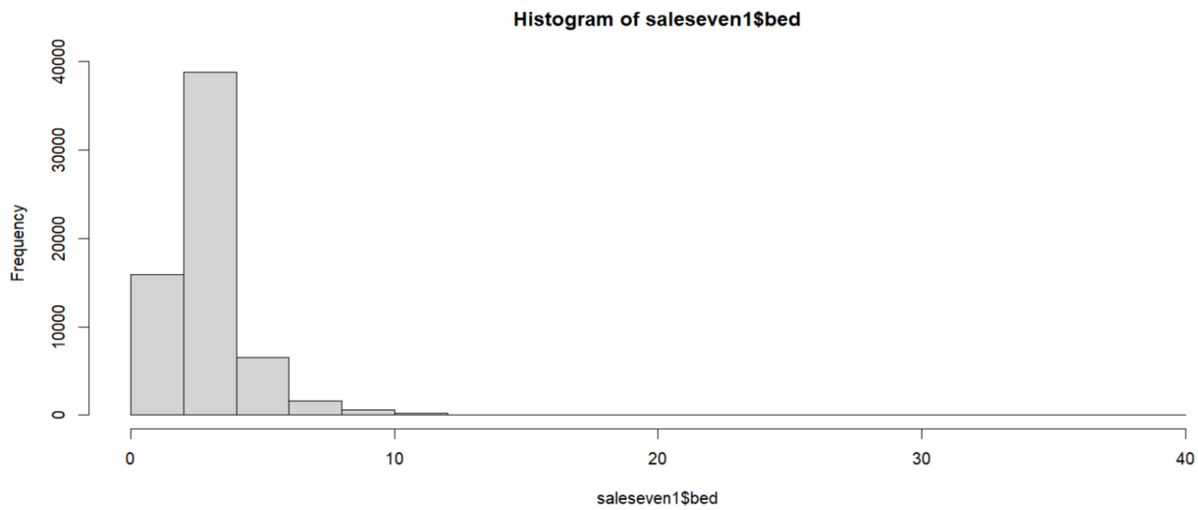


d) saleseven$breakeven

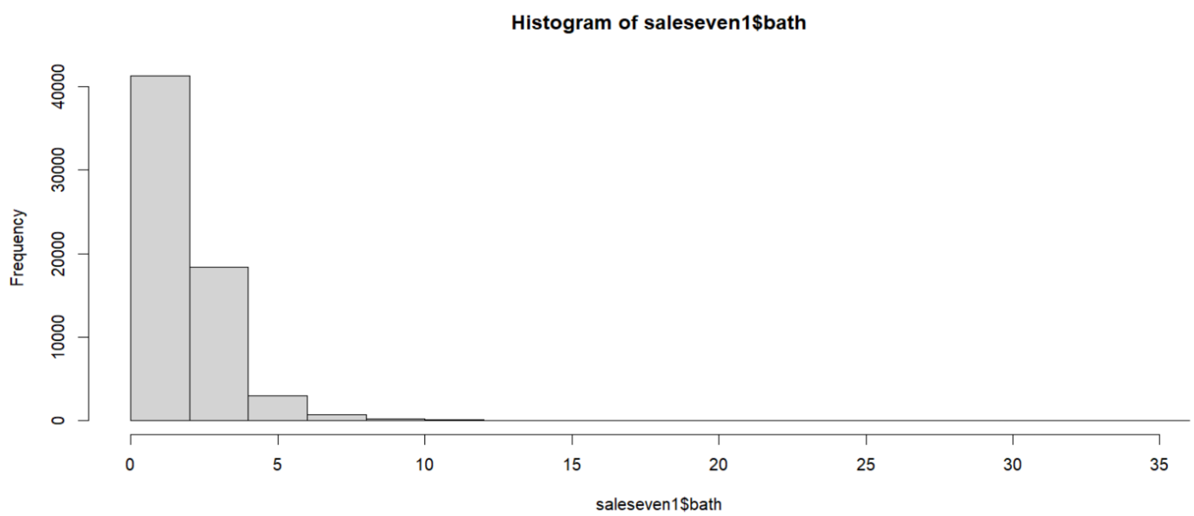


Histograms:

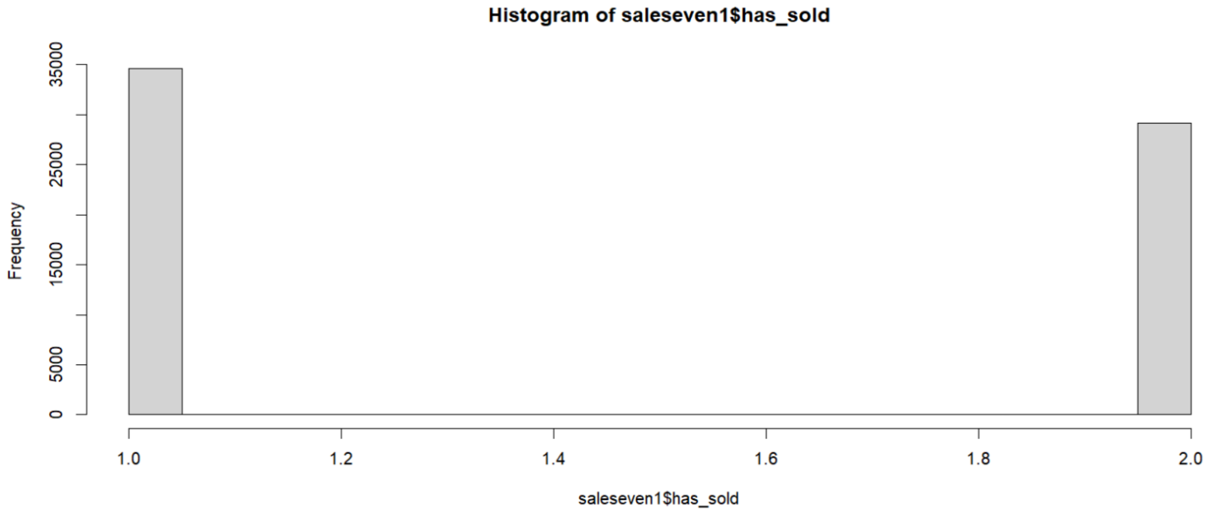
a) saleseven1$bed



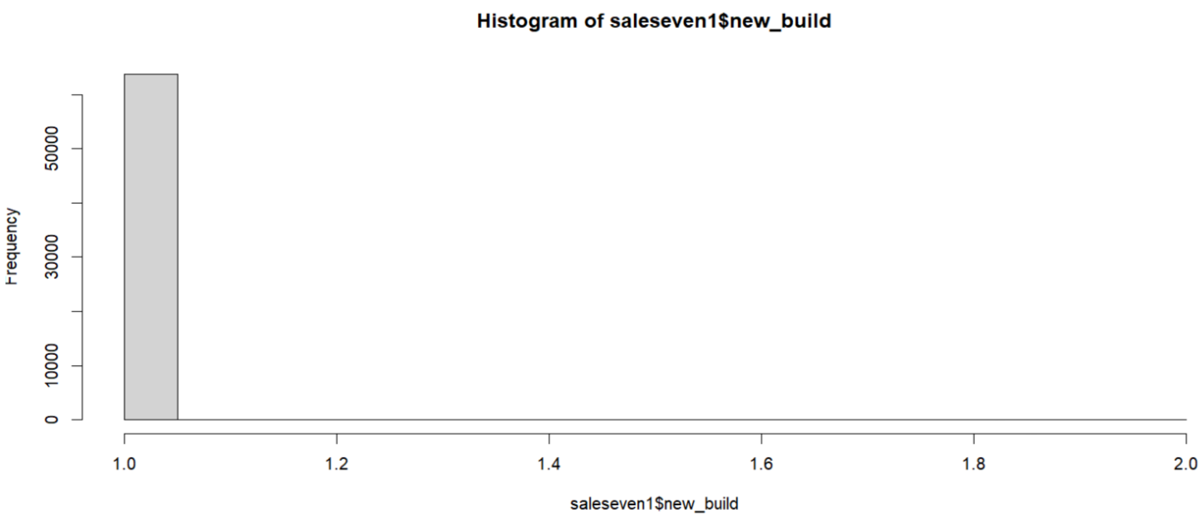
b) saleseven1$bath



c) saleseven1$has_sold



d) saleseven1$new_build



6.3 Transformations:

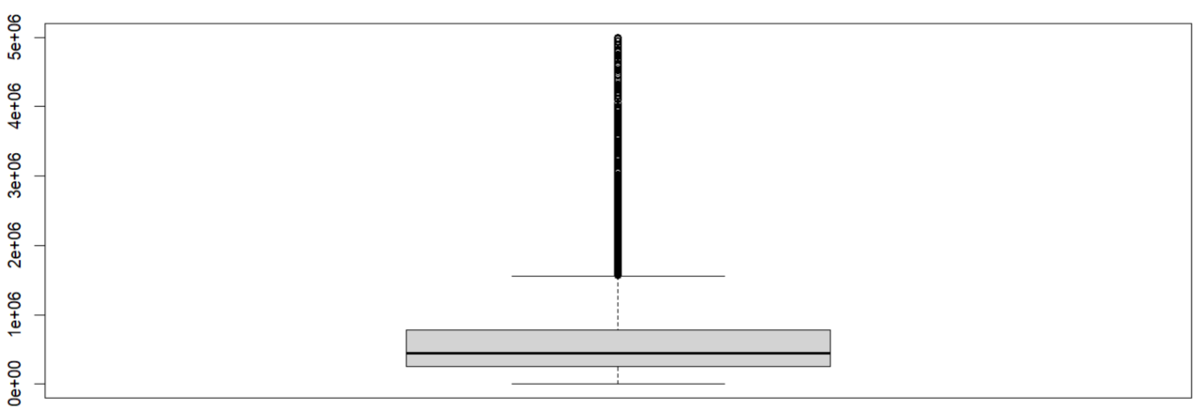
After removing duplicates, and based on summary statistics, box plots and histograms, the following finalized descriptive statistics were generated:

```
status                price               bed                bath              acre_l
ot        full_address
 Length:53884         Min.   :       0   Min.    : 1.000   Min.    : 1.000   Min.
:0.0000   Length:53884
 Class :character     1st Qu.: 265000   1st Qu.: 2.000   1st Qu.: 2.000   1st Q
u.:0.1700   Class :character
 Mode  :character     Median : 450000   Median : 3.000   Median : 2.000   Media
n :0.5000   Mode  :character
                      Mean   : 666906   Mean   : 3.205   Mean   : 2.332   Mean
:0.5476
```

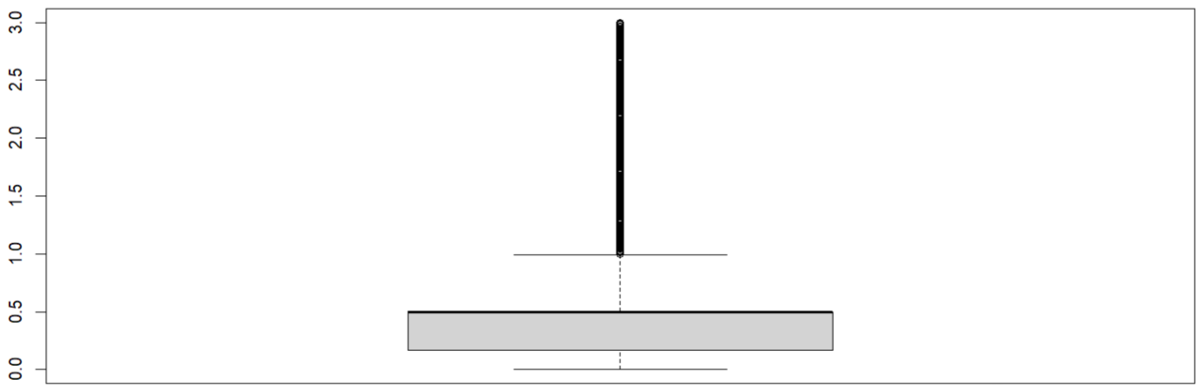
```
                    3rd Qu.: 769000   3rd Qu.: 4.000   3rd Qu.: 3.000   3rd Q
u.:0.5000
                    Max.   :4999999   Max.   :10.000   Max.   :10.000   Max.
:3.0000
    street             city             state             house_size    sol
d_date          zipcode
 Length:53884       Length:53884       Length:53884       Min.   : 100   Leng
th:53884       Length:53884
 Class :character   Class :character   Class :character   1st Qu.:1302   Clas
s :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Median :1744   Mode
:character   Mode  :character
                                                          Mean   :1830
                                                          3rd Qu.:2007
                                                          Max.   :6000
      lat              lng            breakeven        has_sold       new_build
 Min.   :17.98   Min.   :-81.50   Min.   : 1.00   Min.   :1.000   Min.   :1.0
00
 1st Qu.:40.74   1st Qu.:-73.99   1st Qu.:16.50   1st Qu.:1.000   1st Qu.:1.0
00
 Median :41.38   Median :-73.00   Median :25.00   Median :1.000   Median :1.0
00
 Mean   :40.86   Mean   :-72.50   Mean   :29.43   Mean   :1.499   Mean   :1.0
01
 3rd Qu.:42.13   3rd Qu.:-71.41   3rd Qu.:37.00   3rd Qu.:2.000   3rd Qu.:1.0
00
 Max.   :47.32   Max.   :-65.28   Max.   :87.50   Max.   :2.000   Max.   :2.0
00
```

<u>Boxplots</u>

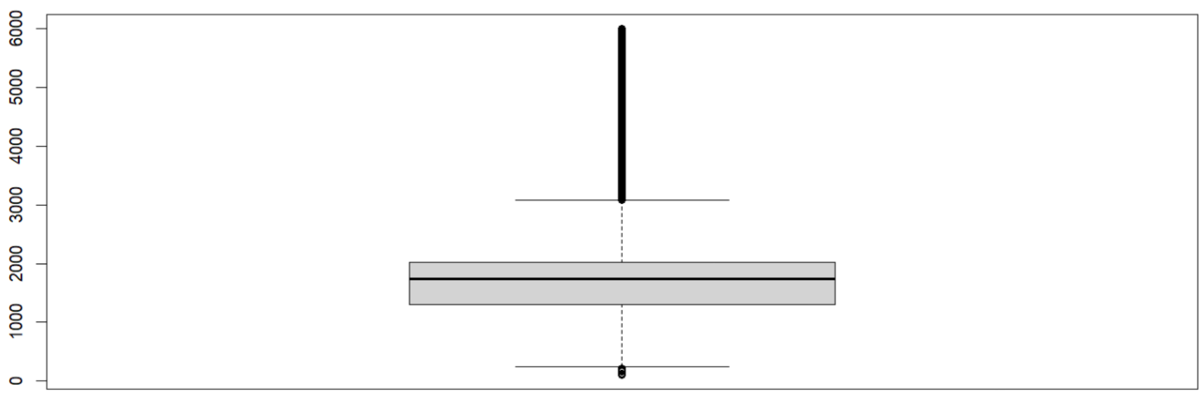
a) saleseven$price (removed data above 5million)



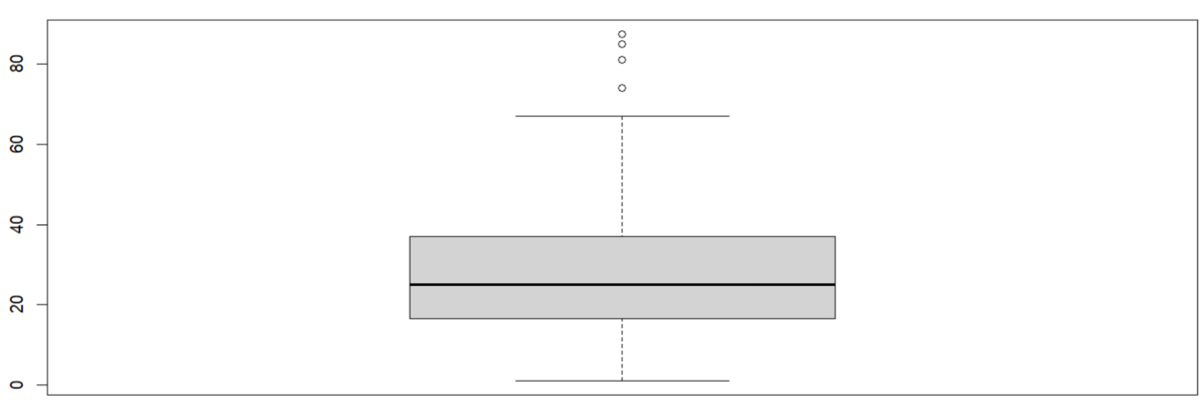
b) saleseven$acre_lot (removing data above 3 acre lots)



c) saleseven$house_size (removing data above 6000)

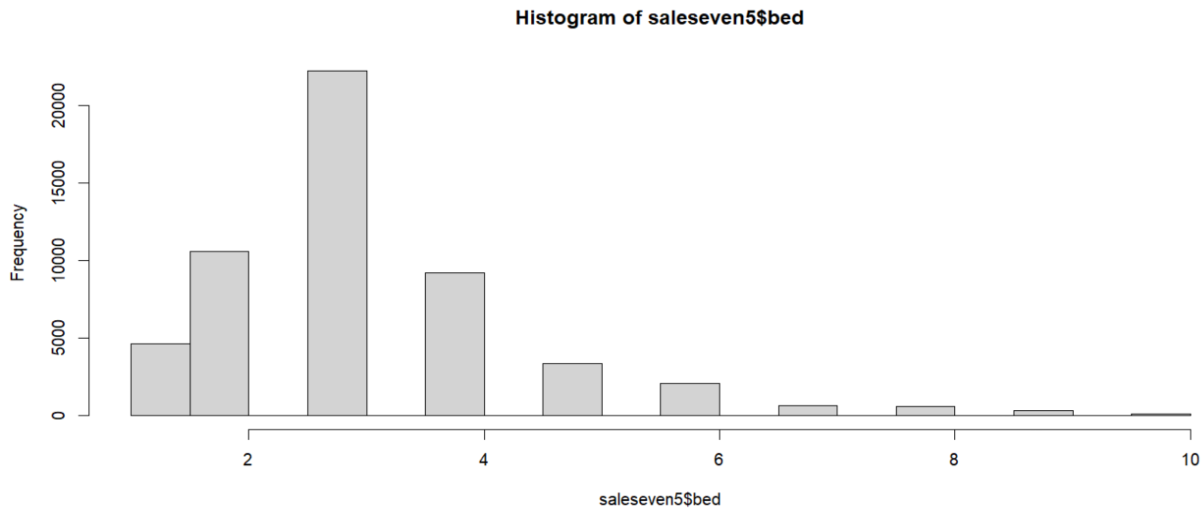


d) saleseven$breakeven (nothing removed)



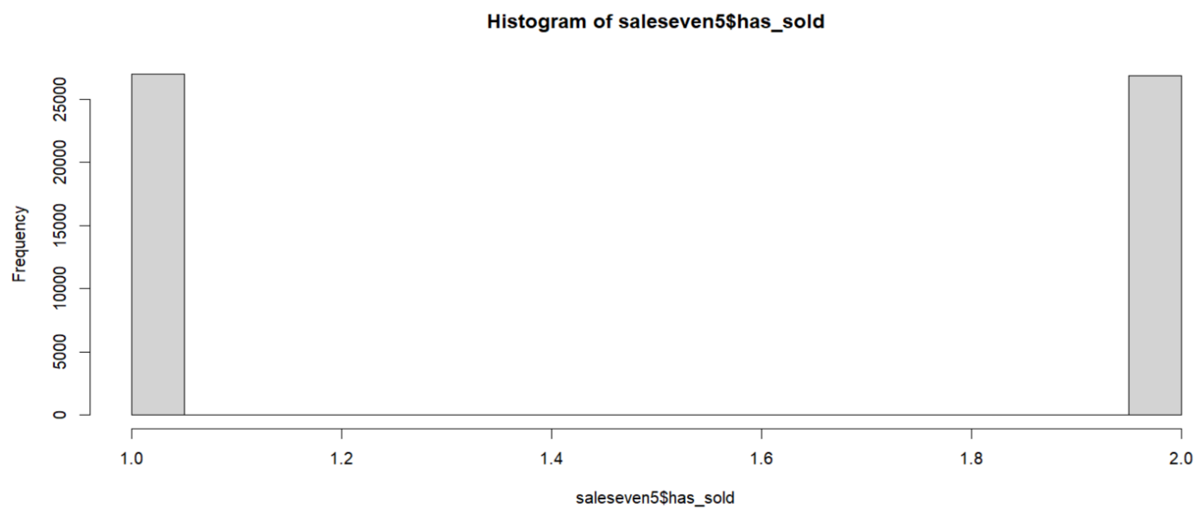
Histogram:

a) saleseven$bed (removing data greater than 10)

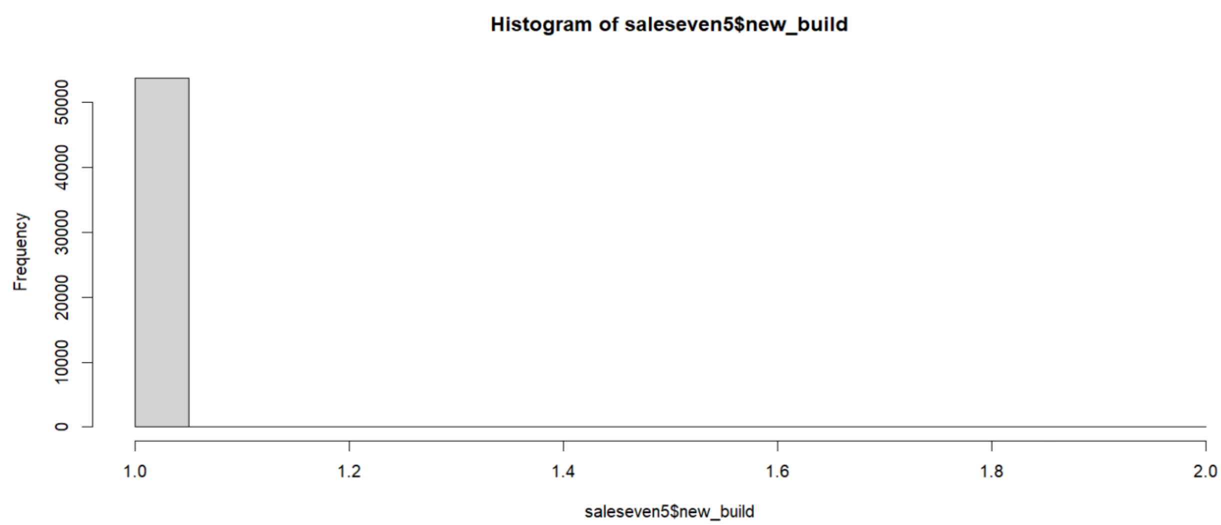


b) saleseven$bath (removing data greater than 10)

c) saleseven$has_sold (not removed)



d) saleseven$new_build (not removed)



For each of the transformations, the main idea was to get as even of a distribution as possible with few leverage points/outliers, while minimizing the amount of lost data. This process was accomplished by using trial and error, iteratively changing the domain of acceptable data, comparing the amount of lost data to the change in distribution. The above transformations were deemed to be the best in regards to this tradeoff.

6.4 Normality of Price:

While, according to ISLR (2021) linear regression assumes the dependent variable to be normally distributed, though due to robust nature of model is not required. However, and Anderson-Darling test was run on both pre-transformed and transformed data:

Pre-Transformed:

```
        Anderson-Darling normality test

data:  salesevenl$price
A = 11661, p-value < 2.2e-16
```

Post-Transformed:

```
        Anderson-Darling normality test

data:  saleseven2$price
A = 5293, p-value < 2.2e-16
```

The A-D test showed that the dependent variable was not normally distributed. While--as noted--the linear regression is robust enough to handle non-normal dependent variables, this test was run to ensure normality. The non-normal nature of price (the dependent variable) should be noted, and is discussed in the limitations section of this report.

Section 7: Model Training and Testing

Before modelling:

1. Dummy variables for is_built and has_sold were dummy coded 0 for no 1 for yes. The 1 and 2 coding was needed for creating histograms. Dummy 0/1 binary coding was needed for modelling.
2. Library caret was then employed to conduct test/training splits (80% train, 20% test). 2 sets of test/train were created trainset1/testset1 for predicting price, and trainset2/testset2 for classifying buy/not buy. A random seed was set to ensure reproducible research.

<u>Model Approximation</u>

In addition to helping determine models for predicting price and classifying homes that have sold/not sold, this research will critically examine which models best answer these questions. In particular, parametric models with defined functions will be used and compared to non-parametric tree structures. All of these models are models used frequently, per literature review (see section 2), when researching the home sales and real estate sectors.

All information about models comes from ISLR-James et al., 2021.

Models were run on two training sets: trainset1 for predicting price, and trainset2 for classification of sold.

<u>Model Testing/Validation</u>

All models were then tested by using testset1 for predicting price, and testset2 for classification to a) gain a prediction from the models created on train sets and b) to calculate mean RSS for prediction of price or confusion matrix and misclassification rate for predicting sold.

This section details both the model training and testing outcomes.

7.1 Predicting Price

Using the literature (see section 2) as a reference, the following models were conducted:

Parametric

1. Linear Regression: used as a base line for research. Linear Model Used to predict price.
2. Ridge and Lasso Regression: multicollinearity amongst predictors arose, necessitating penalty to be applied to features to ensure accurate linear model.
3. Generalized Additive Model: linear model with flexible smoothing, used to see if more flexible approach then pure linear model better predicts price.

Non-Parametric:

1. Decision Tree: Regression tree used to find model for expected price based on features
2. Random Forest: regression random forest using bootstrap method used to find best tree averaging multiple trees with subset features against one another

Mixed Model:

1. Geocoded data of home price, by latitude and longitude plotted to see patterns on sale prices.

7.1.1 Regression:

The regression model is a parametric model of the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon,$$

(James, 2021, p. 82).

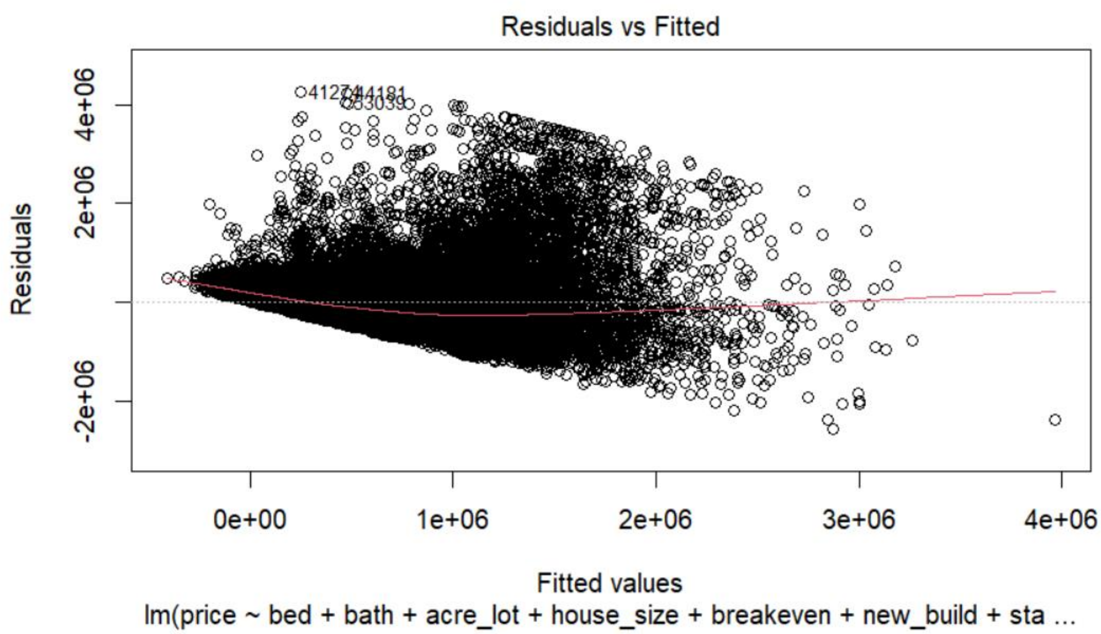
The model is a separable model that takes in beta features sums each up and outputs a predicted value. Here each beta value represents a variation in Y based on 1 unit input in a given feature.

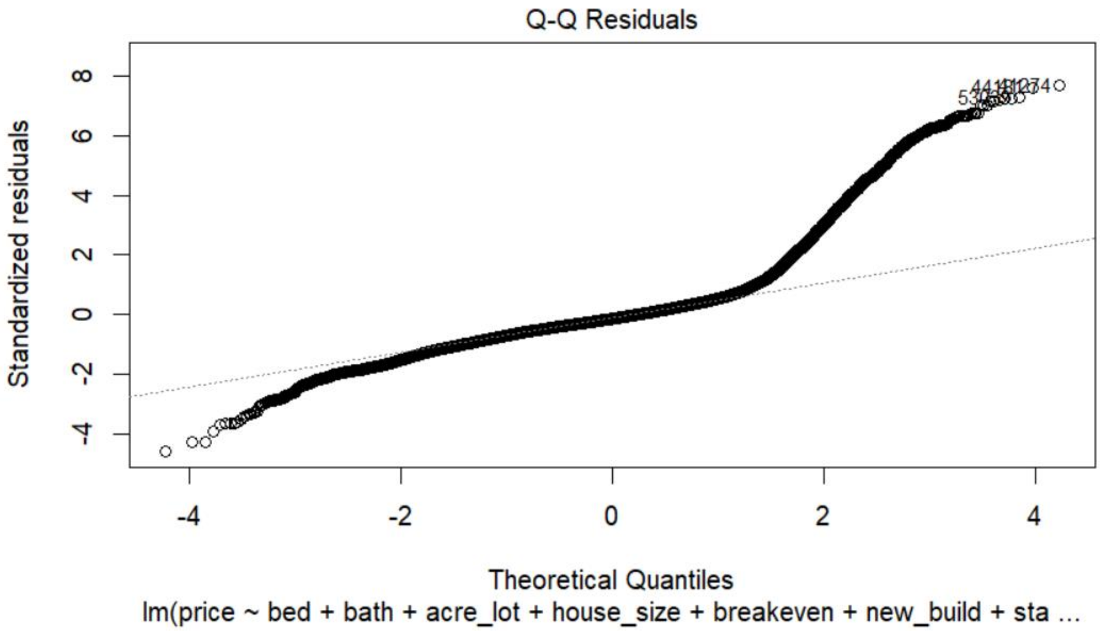
For this research, the following linear regression model was run. To ensure best fit some geographic area needed to be included as dummy variables. To ensure that not too many variables were included, each state was included as a dummy variable (1 is address in state 0 is address not in state):

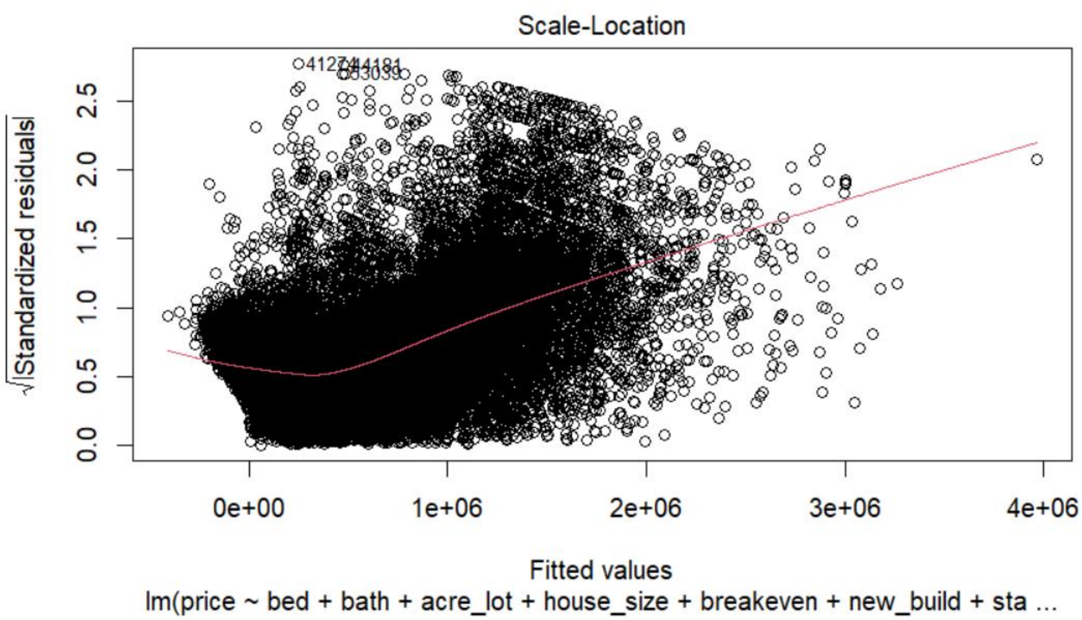
```
Call:
lm(formula = price ~ bed + bath + acre_lot + house_size + breakeven +
    new_build + state_pr + state_ma + state_ct + state_nj + state_nh +
    state_vt + state_ny + state_ri + state_va + state_me + state_pa +
    state_wv, data = trainset1)

Residuals:
     Min       1Q   Median       3Q      Max
-2551892  -278013   -77743   156441  4251715

Coefficients: (1 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.841e+05  5.535e+05  -0.874  0.38186
bed         -9.058e+04  2.502e+03 -36.203  < 2e-16 ***
bath         2.936e+05  3.415e+03  85.982  < 2e-16 ***
acre_lot    -3.197e+04  5.193e+03  -6.156 7.52e-10 ***
house_size   1.442e+02  4.415e+00  32.660  < 2e-16 ***
breakeven    4.528e+03  1.754e+02  25.822  < 2e-16 ***
new_build    2.639e+05  8.373e+04   3.152  0.00162 **
state_pr     1.357e+05  5.537e+05   0.245  0.80634
```

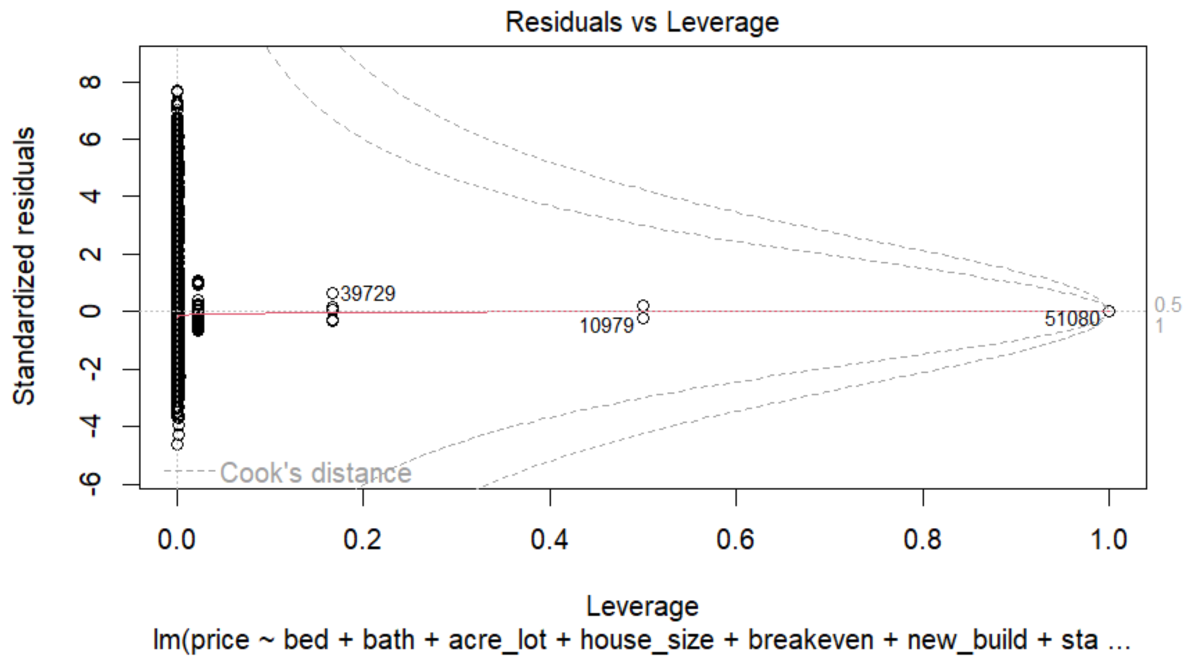
```
state_ma     4.130e+05  5.536e+05   0.746  0.45558
state_ct     8.273e+04  5.535e+05   0.149  0.88119
state_nj     3.216e+05  5.535e+05   0.581  0.56124
state_nh     1.854e+05  5.536e+05   0.335  0.73767
state_vt     2.825e+04  5.537e+05   0.051  0.95931
state_ny     9.400e+05  5.535e+05   1.698  0.08948 .
state_ri     2.423e+05  5.536e+05   0.438  0.66157
state_va    -3.477e+04  6.779e+05  -0.051  0.95909
state_me     1.230e+05  5.536e+05   0.222  0.82424
state_pa     2.835e+05  5.979e+05   0.474  0.63539
state_wv            NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 553500 on 43090 degrees of freedom
Multiple R-squared:  0.3905,   Adjusted R-squared:  0.3903
F-statistic:  1624 on 17 and 43090 DF,  p-value: < 2.2e-16
```

And plots generated:




Q-Q Residuals
Standardized residuals
Theoretical Quantiles
lm(price ~ bed + bath + acre_lot + house_size + breakeven + new_build + sta ...


Scale-Location
√|Standardized residuals|
Fitted values
lm(price ~ bed + bath + acre_lot + house_size + breakeven + new_build + sta ...



A prediction on the test set was run and the mean RSS recorded.

lmfit.prediction <- predict(lmfit, testset1)

meanrss.lm <- mean((lmfit.prediction-testset1$price)^2)

#302373223135 mean rss

As can be seen from the outputs, the adjusted R squared of 0.3903 and the Q-Q and residual plots shows a weak fit, with heteroscedastic residuals. As such, this model, while a good starting point, was deemed insufficient for predicting price.

7.1.2 Regression with Best Subset

The overall model was statistically significant and all features except new_build and states were statistically significant. Next, models were rerun without the non-statistically significant predictors, but the fits did not get better. Given the weak accuracy of the linear regression, the next step was to test subsets of predictors to see if too much bias has been introduced into the model. Per James, et al. (2021, Chapter 6), a best subset selection is often used to create the models with the best combination of features.

The best model removed new_build from the model. Below are the outputs:

Call: regfit.full <- regsubsets(price~bed+bath+acre_lot+house_size+breakeven+new_build+state_pr+state_ma+state_ct+state_nj+state_nh+ state_vt+state_ny+state_ri+state_va+state_me+state_pa+state_wv, testset1)

```
> reg.summary$adjr2
[1] 0.1795277 0.3327589 0.3538804 0.3606908 0.3702281 0.3746100 0.3825219 0.3
843957 0.3850769


Call:
lm(formula = price ~ bed + bath + acre_lot + house_size + breakeven + 
    state_pr + state_ma + state_ct + state_nj + state_nh + state_vt + 
    state_ny + state_ri + state_va + state_me + state_pa + state_wv, 
    data = trainset1)

Residuals:
     Min       1Q   Median       3Q      Max 
-2551950  -278347   -77648   156586  4251497 

Coefficients: (1 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -4.840e+05  5.536e+05  -0.874   0.3820    
bed         -9.075e+04  2.502e+03 -36.275  < 2e-16 ***
bath         2.933e+05  3.414e+03  85.916  < 2e-16 ***
acre_lot    -3.213e+04  5.193e+03  -6.186 6.22e-10 ***
house_size   1.449e+02  4.409e+00  32.873  < 2e-16 ***
breakeven    4.521e+03  1.754e+02  25.778  < 2e-16 ***
state_pr     1.360e+05  5.537e+05   0.246   0.8060    
state_ma     4.145e+05  5.536e+05   0.749   0.4540    
state_ct     8.315e+04  5.536e+05   0.150   0.8806    
state_nj     3.221e+05  5.536e+05   0.582   0.5607    
state_nh     1.854e+05  5.537e+05   0.335   0.7377    
state_vt     2.849e+04  5.538e+05   0.051   0.9590    
state_ny     9.400e+05  5.536e+05   1.698   0.0895 .  
state_ri     2.424e+05  5.537e+05   0.438   0.6616    
state_va    -3.470e+04  6.780e+05  -0.051   0.9592    
state_me     1.231e+05  5.537e+05   0.222   0.8240    
state_pa     2.828e+05  5.980e+05   0.473   0.6363    
state_wv            NA         NA      NA       NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 553500 on 43091 degrees of freedom
Multiple R-squared:  0.3904,	Adjusted R-squared:  0.3901 
F-statistic:  1724 on 16 and 43091 DF,  p-value: < 2.2e-16
```

Again, the model was statistically significant, with the same features being significant. Comparing the adjusted R squared to the linear regression, `0.3901` for the best subset versus `0.3903` showed virtually no improvement.

7.1.3 Ridge and Lasso Regressions

Given the lack of improvement in models between best-subset from the standard linear regression, it became imperative to check the features, to see if features are correlated. Such correlations can reduce the accuracy of the model (James et al., 2021, Chapter 6).

As such, a correlation matrix was plotted:

```
> rcorr(as.matrix(corr_dataframe))
             bed  bath acre_lot house_size breakeven new_build state_pr state
_ma state_ct state_nj state_nh state_vt
bed         1.00  0.61    -0.06       0.58     -0.04     -0.01     0.03     0
.00     0.04     0.11    -0.02     0.00
```

```
bath         0.61  1.00     0.05       0.65     -0.04     -0.01    -0.01     0
.01     0.06     0.11    -0.02    -0.01
acre_lot   -0.06  0.05     1.00       0.13      0.04      0.00    -0.06     0
.00     0.11    -0.20     0.12     0.11
house_size  0.58  0.65     0.13       1.00     -0.01      0.03    -0.03     0
.06     0.06    -0.03     0.02     0.02
breakeven  -0.04 -0.04     0.04      -0.01      1.00      0.00    -0.04     0
.27     0.41    -0.26    -0.16     0.10
new_build  -0.01 -0.01     0.00       0.03      0.00      1.00    -0.01     0
.05    -0.01     0.00    -0.01    -0.01
state_pr    0.03 -0.01    -0.06      -0.03     -0.04     -0.01     1.00    -0
.08    -0.10    -0.09    -0.04    -0.03
state_ma    0.00  0.01     0.00       0.06      0.27      0.05    -0.08     1
.00    -0.23    -0.21    -0.10    -0.08
state_ct    0.04  0.06     0.11       0.06      0.41     -0.01    -0.10    -0
.23     1.00    -0.25    -0.12    -0.09
state_nj    0.11  0.11    -0.20      -0.03     -0.26      0.00    -0.09    -0
.21    -0.25     1.00    -0.10    -0.08
state_nh   -0.02 -0.02     0.12       0.02     -0.16     -0.01    -0.04    -0
.10    -0.12    -0.10     1.00    -0.04
state_vt    0.00 -0.01     0.11       0.02      0.10     -0.01    -0.03    -0
.08    -0.09    -0.08    -0.04     1.00
state_ny   -0.15 -0.11    -0.11      -0.11     -0.28     -0.02    -0.10    -0
.23    -0.28    -0.24    -0.11    -0.09
state_ri    0.03 -0.02    -0.05       0.04     -0.06     -0.01    -0.05    -0
.11    -0.13    -0.12    -0.05    -0.04
state_va    0.00  0.00     0.00       0.00      0.00      0.00     0.00     0
.00     0.00     0.00     0.00     0.00
state_me   -0.01 -0.06     0.25      -0.01     -0.08     -0.01    -0.05    -0
.11    -0.13    -0.12    -0.05    -0.04
state_pa    0.01  0.03     0.01       0.04     -0.01      0.00     0.00    -0
.01    -0.01    -0.01     0.00     0.00
state_wv    0.00  0.00     0.00       0.00      0.00      0.00     0.00     0
.00     0.00     0.00     0.00     0.00
           state_ny state_ri state_va state_me state_pa state_wv
bed           -0.15     0.03        0    -0.01     0.01        0
bath          -0.11    -0.02        0    -0.06     0.03        0
acre_lot      -0.11    -0.05        0     0.25     0.01        0
house_size    -0.11     0.04        0    -0.01     0.04        0
breakeven     -0.28    -0.06        0    -0.08    -0.01        0
new_build     -0.02    -0.01        0    -0.01     0.00        0
state_pr      -0.10    -0.05        0    -0.05     0.00        0
state_ma      -0.23    -0.11        0    -0.11    -0.01        0
state_ct      -0.28    -0.13        0    -0.13    -0.01        0
state_nj      -0.24    -0.12        0    -0.12    -0.01        0
state_nh      -0.11    -0.05        0    -0.05     0.00        0
state_vt      -0.09    -0.04        0    -0.04     0.00        0
state_ny       1.00    -0.13        0    -0.13    -0.01        0
state_ri      -0.13     1.00        0    -0.06     0.00        0
state_va       0.00     0.00        1     0.00     0.00        0
state_me      -0.13    -0.06        0     1.00     0.00        0
state_pa      -0.01     0.00        0     0.00     1.00        0
state_wv       0.00     0.00        0     0.00     0.00        1

n= 43108


P
           bed    bath   acre_lot house_size breakeven new_build state_pr sta
te_ma state_ct state_nj state_nh state_vt
bed               0.0000 0.0000   0.0000     0.0000    0.0802    0.0000   0.4
042   0.0000   0.0000   0.0000   0.8139
bath       0.0000        0.0000   0.0000     0.0000    0.0511    0.0207   0.0
462   0.0000   0.0000   0.0000   0.0825
```

```
acre_lot   0.0000 0.0000         0.0000     0.0000    0.5676    0.0000   0.3
220   0.0000   0.0000   0.0000   0.0000
house_size 0.0000 0.0000 0.0000             0.0074    0.0000    0.0000   0.0
000   0.0000   0.0000   0.0000   0.0000
breakeven  0.0000 0.0000 0.0000   0.0074              0.5809    0.0000   0.0
000   0.0000   0.0000   0.0000   0.0000
new_build  0.0802 0.0511 0.5676   0.0000    0.5809              0.2006   0.0
000   0.0384   0.7121   0.1461   0.2550
state_pr   0.0000 0.0207 0.0000   0.0000    0.0000    0.2006             0.0
000   0.0000   0.0000   0.0000   0.0000
state_ma   0.4042 0.0462 0.3220   0.0000    0.0000    0.0000    0.0000
0.0000   0.0000   0.0000   0.0000
state_ct   0.0000 0.0000 0.0000   0.0000    0.0000    0.0384    0.0000   0.0
000            0.0000   0.0000   0.0000
state_nj   0.0000 0.0000 0.0000   0.0000    0.0000    0.7121    0.0000   0.0
000   0.0000            0.0000   0.0000
state_nh   0.0000 0.0000 0.0000   0.0000    0.0000    0.1461    0.0000   0.0
000   0.0000   0.0000            0.0000
state_vt   0.8139 0.0825 0.0000   0.0000    0.0000    0.2550    0.0000   0.0
000   0.0000   0.0000   0.0000
state_ny   0.0000 0.0000 0.0000   0.0000    0.0000    0.0006    0.0000   0.0
000   0.0000   0.0000   0.0000   0.0000
state_ri   0.0000 0.0002 0.0000   0.0000    0.0000    0.1020    0.0000   0.0
000   0.0000   0.0000   0.0000   0.0000
state_va   0.8454 0.6776 0.7197   0.8898    0.9169    0.9639    0.7850   0.5
353   0.4524   0.5069   0.7568   0.8083
state_me   0.0641 0.0000 0.0000   0.0041    0.0000    0.0991    0.0000   0.0
000   0.0000   0.0000   0.0000   0.0000
state_pa   0.0057 0.0000 0.1857   0.0000    0.1104    0.9376    0.6366   0.2
829   0.1930   0.2503   0.5916   0.6744
state_wv   0.5800 0.7688 0.5039   0.9704    0.3888    0.9745    0.8471   0.6
611   0.5952   0.6389   0.8266   0.8638
           state_ny state_ri state_va state_me state_pa state_wv
bed        0.0000   0.0000   0.8454   0.0641   0.0057   0.5800
bath       0.0000   0.0002   0.6776   0.0000   0.0000   0.7688
acre_lot   0.0000   0.0000   0.7197   0.0000   0.1857   0.5039
house_size 0.0000   0.0000   0.8898   0.0041   0.0000   0.9704
breakeven  0.0000   0.0000   0.9169   0.0000   0.1104   0.3888
new_build  0.0006   0.1020   0.9639   0.0991   0.9376   0.9745
state_pr   0.0000   0.0000   0.7850   0.0000   0.6366   0.8471
state_ma   0.0000   0.0000   0.5353   0.0000   0.2829   0.6611
state_ct   0.0000   0.0000   0.4524   0.0000   0.1930   0.5952
state_nj   0.0000   0.0000   0.5069   0.0000   0.2503   0.6389
state_nh   0.0000   0.0000   0.7568   0.0000   0.5916   0.8266
state_vt   0.0000   0.0000   0.8083   0.0000   0.6744   0.8638
state_ny            0.0000   0.4628   0.0000   0.2034   0.6036
state_ri   0.0000            0.7275   0.0000   0.5461   0.8054
state_va   0.4628   0.7275            0.7253   0.9867   0.9946
state_me   0.0000   0.0000   0.7253            0.5427   0.8038
state_pa   0.2034   0.5461   0.9867   0.5427            0.9906
state_wv   0.6036   0.8054   0.9946   0.8038   0.9906
```

As can be seen, many of the features are correlated, such as bed and bath (above a value of .5), with p values indicating that they are statistically significant. As a result, two shrinkage methods the ridge regression and the lasso regression were run.

7.1.3(a) Ridge Regression:

The idea behind the ridge regression is to apply a shrinkage penalty to the predictor variables in order to remove the problem of multicollinearity.

The ridge regression given:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2,$$

(James et al., 2021, p. 237)

Using the shrinkage parameter λ, the idea is to shrink each the beta coefficients (with the exception of the intercept), reducing variance, in turn reducing correlated values.

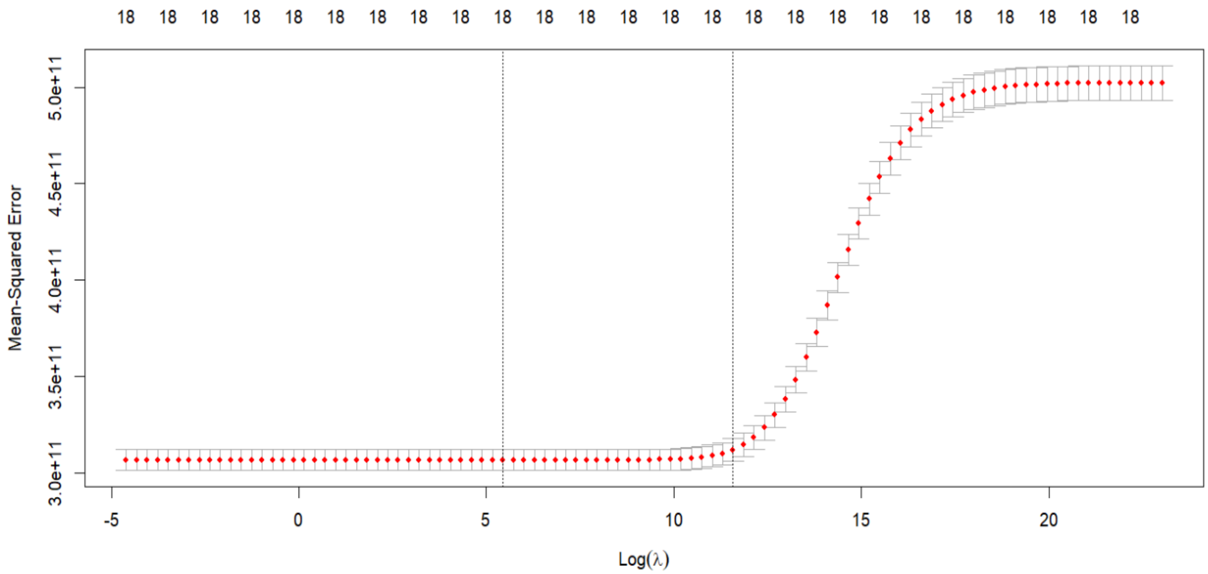
Below are the outputs from the ridge regression, with:

1. cross validation to find the optimal λ value:
2. the model run with optimal λ
3. predictions run on the test set
4. predictions used to calculate mean RSS

Call: ridge.mod <- cv.glmnet(x_training,y_training, alpha = 0, lambda = grid, parallel = TRUE)

Summary:

```
> summary(ridge.mod)
           Length Class  Mode
lambda     100    -none- numeric
cvm        100    -none- numeric
cvsd       100    -none- numeric
cvup       100    -none- numeric
cvlo       100    -none- numeric
nzero      100    -none- numeric
call         6    -none- call
name         1    -none- character
glmnet.fit  12    elnet  list
lambda.min   1    -none- numeric
lambda.1se   1    -none- numeric
index        2    -none- numeric
```




```
> lambda.best
[1] 231.013
```

```
> summary(ridge.prediction)
       s1
 Min.   :-250733
 1st Qu.: 330367
 Median : 601282
 Mean   : 665910
 3rd Qu.: 946012
 Max.   :3316953
```

Mean RSS from Predictions (using test set data):

```
> meanrss.ridge
[1] 302368819401
```

Coefficients produced from training set:

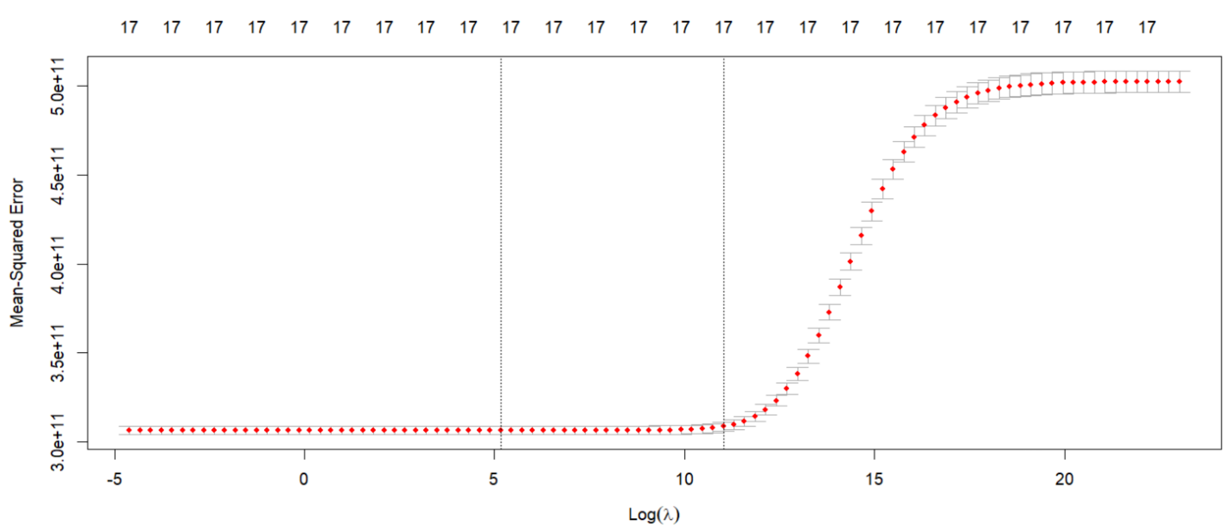
```
> best.model <- glmnet(x_test, y_test, alpha = 0, lambda = lambda.best)
> coef(best.model)
20 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  -53777.8877
(Intercept)       .
bed          -72764.4490
bath         292915.8587
acre_lot     -31315.5568
house_size      126.5457
breakeven      4356.0326
new_build    114558.9253
state_pr    -352848.1748
state_ma     -56781.7450
```

```
state_ct     -355386.0359
state_nj     -137755.9952
state_nh     -252554.0352
state_vt     -422070.6587
state_ny      500088.7932
state_ri     -211018.3657
state_va           .
state_me     -320657.7976
state_pa     -248920.1953
state_wv
```

(Note, state_va and state_wv have dots next to their intercepts because there were few observations, none of which made it into the training set).

To compare to the linear regression, the ridge regression was run without new_build, following previous steps:

Call:
```
> ridge.mod1 <- cv.glmnet(x_training1,y_training1, alpha = 0, lambda =
grid1, parallel = TRUE)
> summary(ridge.mod1)
           Length Class  Mode
lambda     100    -none- numeric
cvm        100    -none- numeric
cvsd       100    -none- numeric
cvup       100    -none- numeric
cvlo       100    -none- numeric
nzero      100    -none- numeric
call         6    -none- call
name         1    -none- character
glmnet.fit  12    elnet  list
lambda.min   1    -none- numeric
lambda.1se   1    -none- numeric
index        2    -none- numeric
```




```
> lambda.best1
[1] 174.7528
```

```
> ridge.prediction1 <- predict(ridge.mod1, s=lambda.best1, newx = x_test1)
> summary(ridge.prediction1)
       s1
 Min.   :-251307
 1st Qu.: 330507
 Median : 601886
 Mean   : 665964
 3rd Qu.: 944804
 Max.   :3318305
```

```
Mean RSS:
> meanrss.ridge1
[1] 302364263316
```

Coefficient Outputs:

```
> best.model1 <- glmnet(x_test1, y_test1, alpha = 0, lambda = lambda.best1)
> coef(best.model1)
19 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  -53468.7612
(Intercept)       .
bed          -72865.5856
bath         292891.8829
acre_lot     -31374.6431
house_size      126.7419
breakeven      4353.2527
state_pr    -353021.2255
state_ma     -56413.4842
state_ct    -355533.5853
state_nj    -137869.8015
state_nh    -252798.4819
state_vt    -422233.8680
state_ny     499898.7163
state_ri    -211240.0633
state_va          .
state_me    -320837.7905
state_pa    -249494.0582
state_wv
```

7.1.3(b) Lasso:

Subsequently, another shrinkage model was used, the Lasso.

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

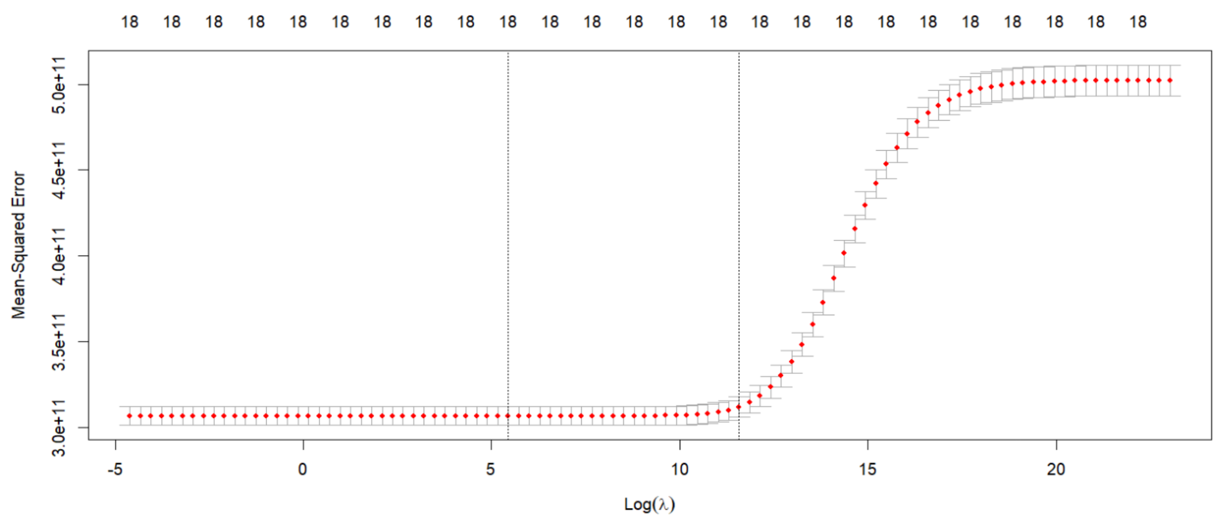
(James et al., 2021, p. 241).

Here, the shrinkage penalty λ is applied with an absolute value of the coefficient, hence resulting in an l1 penalty that allows for variable reduction. Similar to the ridge regression, a lasso was carried out:

1. cross validation to find the optimal λ value:

2. the model run with optimal λ
3. predictions run on the test set
4. predictions used to calculate mean RSS

```
Call: > lasso.mod <- cv.glmnet(x_training,y_training, alpha = 1, lambda = gr
id, parallel = TRUE)

> summary(lasso.mod)
           Length Class  Mode     
lambda     100    -none- numeric  
cvm        100    -none- numeric  
cvsd       100    -none- numeric  
cvup       100    -none- numeric  
cvlo       100    -none- numeric  
nzero      100    -none- numeric  
call         6    -none- call     
name         1    -none- character
glmnet.fit  12    elnet  list     
lambda.min   1    -none- numeric  
lambda.1se   1    -none- numeric  
index        2    -none- numeric  
```




```
> lambda.best.lasso
[1] 43.28761
```

```
> summary(lasso.prediction)
       s1          
 Min.   :-250583  
 1st Qu.: 330390  
 Median : 601203  
 Mean   : 665907  
 3rd Qu.: 946199  
 Max.   :3317370  
```

Mean RSS:

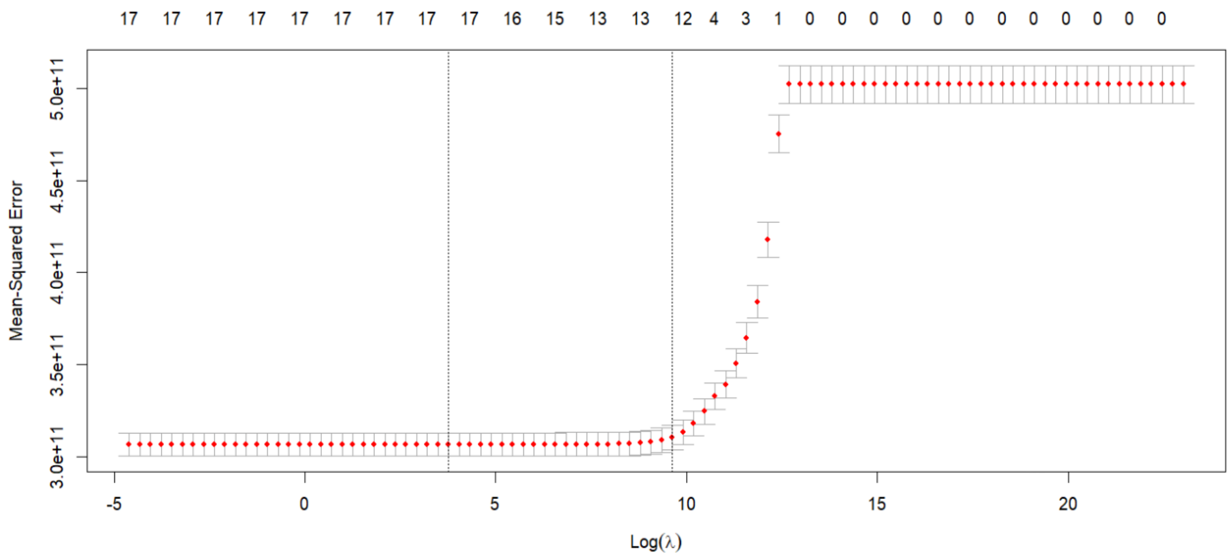
```
> meanrss.lasso
[1] 302366775867
```

Coefficients From Model:

```
> best.model.lasso <- glmnet(x_test, y_test, alpha = 1, lambda = lambda.best.
lasso)
> coef(best.model.lasso)
20 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  -55518.9744
(Intercept)       .
bed          -72773.7471
bath         293032.2097
acre_lot     -31240.7251
house_size      126.4483
breakeven      4357.1753
new_build    113207.5242
state_pr    -351229.7786
state_ma     -55199.7076
state_ct    -353893.9030
state_nj    -136152.1590
state_nh    -250875.2311
state_vt    -420409.3422
state_ny     501752.4104
state_ri    -209282.6061
state_va          .
state_me    -318993.5107
state_pa    -242908.7751
state_wv
```

Similar to the ridge regression, the lasso was re-run without new_build:

```
Call: > lasso.mod1 <- cv.glmnet(x_training1,y_training1, alpha = 1, lambda =
grid1, parallel = TRUE)
> summary(lasso.mod1)
           Length Class  Mode
lambda     100    -none- numeric
cvm        100    -none- numeric
cvsd       100    -none- numeric
cvup       100    -none- numeric
cvlo       100    -none- numeric
nzero      100    -none- numeric
call         6    -none- call
name         1    -none- character
glmnet.fit  12    elnet  list
lambda.min   1    -none- numeric
lambda.1se   1    -none- numeric
index        2    -none- numeric
```




```
> lambda.best.lasso1
[1] 43.28761
```

```
> summary(lasso.prediction1)
       s1
 Min.   :-251044
 1st Qu.: 330525
 Median : 601749
 Mean   : 665962
 3rd Qu.: 944757
 Max.   :3318421
```

Mean RSS:

```
> meanrss.lasso1
[1] 302361386135
```

Coefficients from Output:

```
> best.model.lasso1 <- glmnet(x_test1, y_test1, alpha = 1, lambda = lambda.be
st.lasso1)
> coef(best.model.lasso1)
19 x 1 sparse Matrix of class "dgCMatrix"
                       s0
(Intercept)  -55333.721
(Intercept)       .
bed          -72849.624
bath         292963.144
acre_lot     -31293.993
house_size      126.642
breakeven      4353.389
state_pr    -351223.824
state_ma     -54653.177
state_ct    -353831.622
state_nj    -136088.692
state_nh    -250940.571
state_vt    -420358.626
state_ny     501696.208
state_ri    -209319.475
state_va          .
state_me    -318987.741
```

```
state_pa    -243215.392
state_wv
```

The ridge and lasso both showed improvement on the linear regression (measured by mean RSS—302373223135, linear regression, 302364263316, ridge, 302361386135, lasso). However, the fit still remains weak and further models are to be investigated.

7.1.4 General Additive Model:

Given the poor fits seen so far, it is apparent that the relationship between home attributes and price may not be linear. Rather than guess a polynomial degree or place arbitrary knots, a GAM was employed with smoothing.

GAMs

General Additive Model:

$$\begin{aligned} y_i &= \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i \\ &= \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \epsilon_i. \end{aligned}$$

(James et al., 20201, p. 307).

Here, a non-linear term is multiplied by each coefficient to result in a non-linear smooth term that might produce a better fit. Below are the outputs from the GAM, including: coefficients, predictions, and mean RSS.

GAM Without Smoothing:

```
> gam1 <- gam(price~bed+bath+acre_lot+house_size+breakeven+state_pr+state_ma+
state_ct+state_nj+state_nh+ state_vt+state_ny+state_ri+state_va+state_me+stat
e_pa+state_wv, family = gaussian(), data = trainset1)
> summary(gam1)

Family: gaussian
Link function: identity

Formula:
price ~ bed + bath + acre_lot + house_size + breakeven + state_pr +
    state_ma + state_ct + state_nj + state_nh + state_vt + state_ny +
    state_ri + state_va + state_me + state_pa + state_wv

Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.373e+05  5.572e+04  -4.259 2.06e-05 ***
bed         -9.075e+04  2.502e+03 -36.275  < 2e-16 ***
bath         2.933e+05  3.414e+03  85.916  < 2e-16 ***
acre_lot    -3.213e+04  5.193e+03  -6.186 6.22e-10 ***
house_size   1.449e+02  4.409e+00  32.873  < 2e-16 ***
breakeven    4.521e+03  1.754e+02  25.778  < 2e-16 ***
state_pr    -1.107e+05  5.654e+04  -1.958  0.05019 .
state_ma     1.678e+05  5.544e+04   3.026  0.00248 **
state_ct    -1.636e+05  5.540e+04  -2.953  0.00315 **
state_nj     7.538e+04  5.537e+04   1.361  0.17338
state_nh    -6.127e+04  5.624e+04  -1.089  0.27596
state_vt    -2.182e+05  5.702e+04  -3.827  0.00013 ***
```

```
state_ny     6.933e+05  5.530e+04  12.538  < 2e-16 ***
state_ri    -4.343e+03  5.598e+04  -0.078  0.93816
state_va    -2.814e+05  3.642e+05  -0.773  0.43975
state_me    -1.236e+05  5.605e+04  -2.205  0.02746 *
state_pa     3.609e+04  2.152e+05   0.168  0.86683
state_wv    -2.467e+05  5.122e+05  -0.482  0.63001
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


Rank: 17/18
R-sq.(adj) =   0.39   Deviance explained =   39%
GCV = 3.0652e+11  Scale est. = 3.064e+11  n = 43108
```

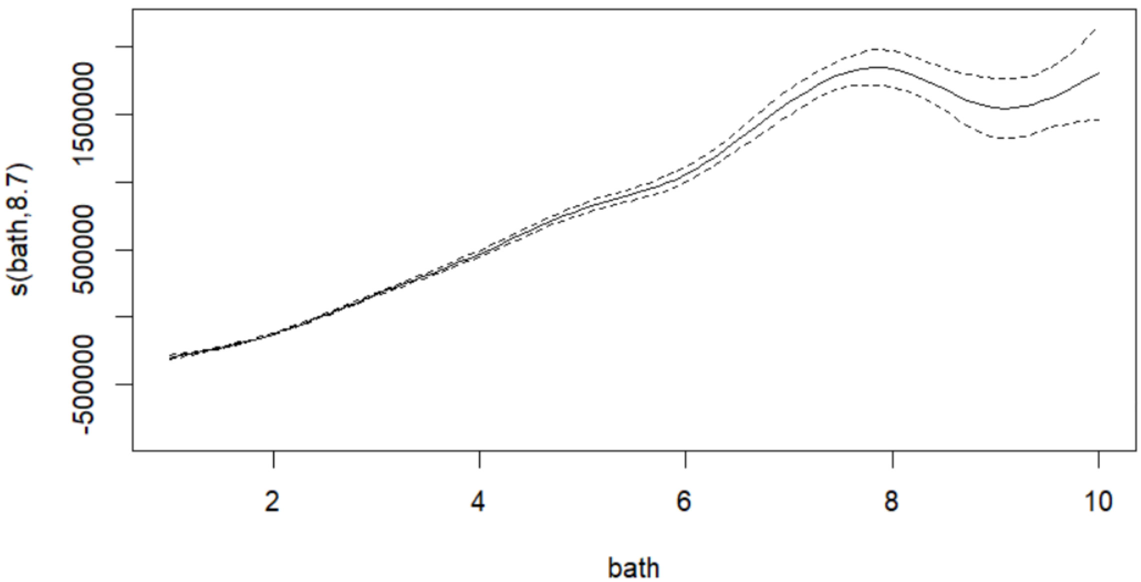
7.1.4 (a) GAM With Smoothing

```
> gam2 <- gam(price~s(bed)+s(bath)+s(acre_lot)+s(house_size)+s(breakeven), fa
mily = gaussian(), data = trainset1)
> summary(gam2)

Family: gaussian
Link function: identity

Formula:
price ~ s(bed) + s(bath) + s(acre_lot) + s(house_size) + s(breakeven)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   667647       2828   236.1   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
                edf Ref.df     F p-value
s(bed)        7.766  8.484 118.1  <2e-16 ***
s(bath)       8.697  8.948 660.5  <2e-16 ***
s(acre_lot)   8.943  8.999 192.7  <2e-16 ***
s(house_size) 7.821  8.609 115.4  <2e-16 ***
s(breakeven)  8.982  9.000 331.6  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.314   Deviance explained = 31.4%
GCV = 3.4517e+11  Scale est. = 3.4483e+11  n = 43108
```

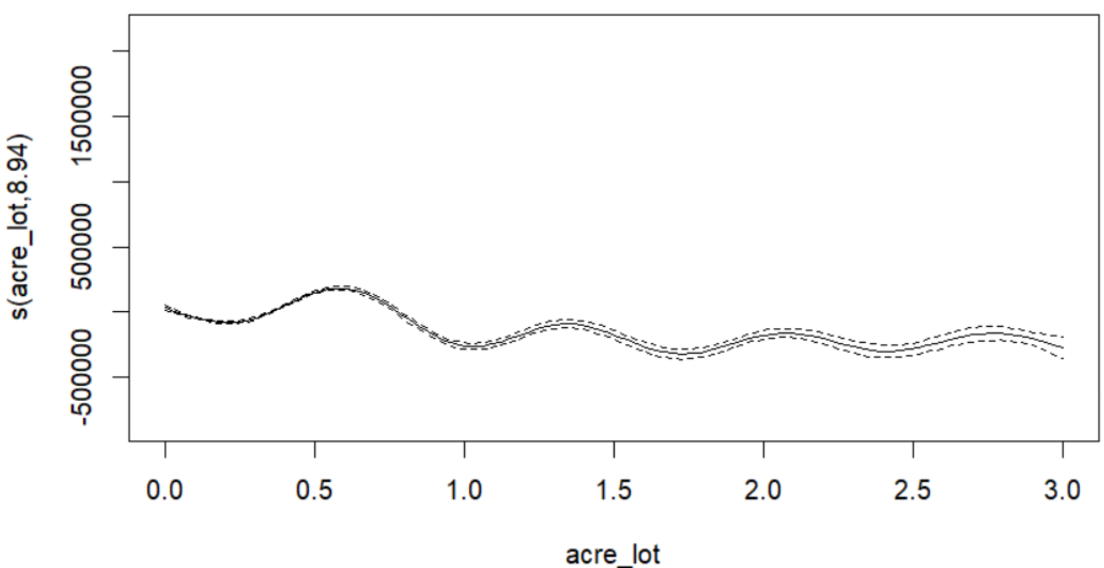
Mean RSS:

```
> meanrss_gam
[1] 342084229665
```

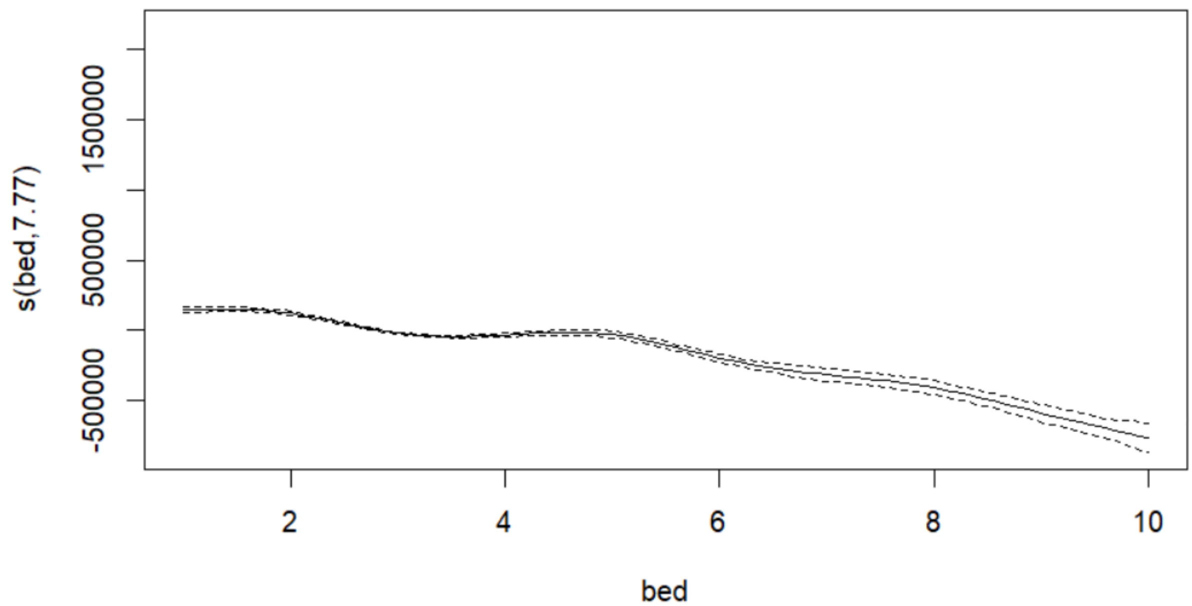
Plots of features to output:



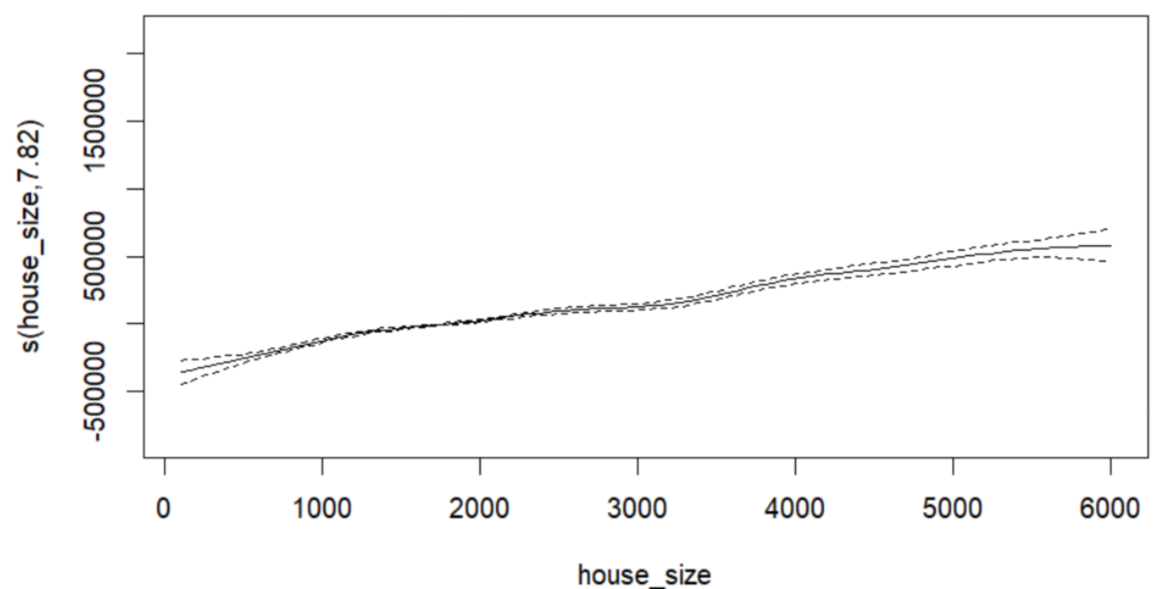
s(bath,8.7)
-500000
500000
1500000
2
4
6
8
10
bath



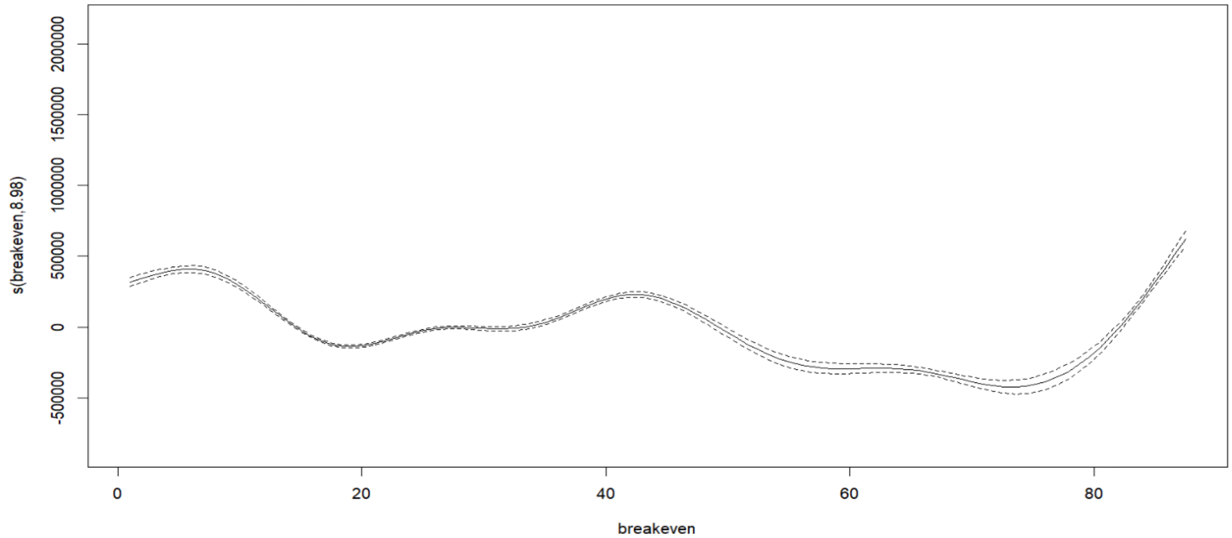
s(acre_lot,8.94)
-500000
500000
1500000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
acre_lot



s(bed,7.77)
-500000
500000
1500000
2
4
6
8
10
bed



s(house_size,7.82)
-500000
500000
1500000
0
1000
2000
3000
4000
5000
6000
house_size



Comparing the RSS of the GAM to the previous models--302373223135, linear regression, 302364263316, ridge, 302361386135, lasso—of 342084229665, it becomes apparent that the fit actually gets worse the less linear the model is. As a result, non-parametric models should be examined next.

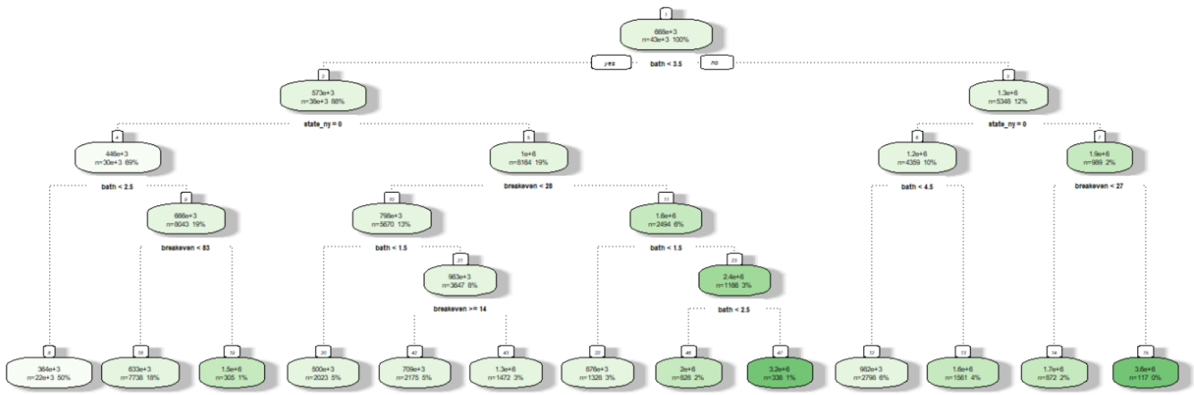
7.1.5 Regression Tree Methods: Decision and Random Forest

7.1.5(a) Decision Tree

Decision Trees are a powerful way of helping to segment feature space, creating non-overlapping regions of predicted average values—price in terms of this research—based on some aspect of the features. These non-parametric models use recursive splitting to create these boundaries, where the key idea is that the boundaries of regions, and subsequent predicted means, are created to minimize mean RSS (James et al., 2021, Chapter 8). Note: due to the small number of observations for Virginia and West Virginia, observations in these two states were removed to ensure accurate predictions.

Below are the outputs of the regression decision tree used in this research, along with calls to predict the values of the test set so as to calculate mean RSS.

```
Call:treereg_train <- rpart(price ~ bed+bath+acre_lot+house_size+breakeven+ne
w_build+state_pr+state_ma+state_ct+state_nj+state_nh+ state_vt+state_ny+state
_ri+state_me+state_pa, data = trainset1)
```



Mean RSS of the predicted values of the test class:

```
 > meanRSS.regtree_rf
[1] 191196646630
```

Compared to previous models--302373223135, linear regression, 302364263316, ridge, 302361386135, lasso, 342084229665, GAM—the decision tree produced a substantially better mean RSS—191196646630—leading to the belief that non-parametric models do better predict home prices.

7.1.5 (b) Random Forest:

Along with a regression decision tree, a regression random forest was run. The idea, according to James et al (2021, Chapter 7), is to run multiple decision trees on a subset of predictors, usually the square root of the numbers of features in the feature set, and then average across them. This in turn results, usually in a more accurate feature segmentation, and better predictions/ lower mean RSS.

Below are the outputs of the random forest, including predictions and mean RSS on test set.

```
> rf_regression <- randomForest(price~bed+bath+acre_lot+house_size+breakeven+
has_sold+new_build+state_pr+state_ma+state_ct+state_nj+state_nh+ state_vt+sta
te_ny+state_ri+state_me+state_pa, data = trainset1, ntree=5)
> rf_regression

Call:
 randomForest(formula = price ~ bed + bath + acre_lot + house_size +      bre
akeven + has_sold + new_build + state_pr + state_ma +      state_ct + state_n
j + state_nh + state_vt + state_ny + state_ri +      state_me + state_pa, dat
a = trainset1, ntree = 5)
               Type of random forest: regression
                     Number of trees: 5
No. of variables tried at each split: 5

          Mean of squared residuals: 236120844179
                    % Var explained: 53


Mean of squared residuals: 236120844179
```

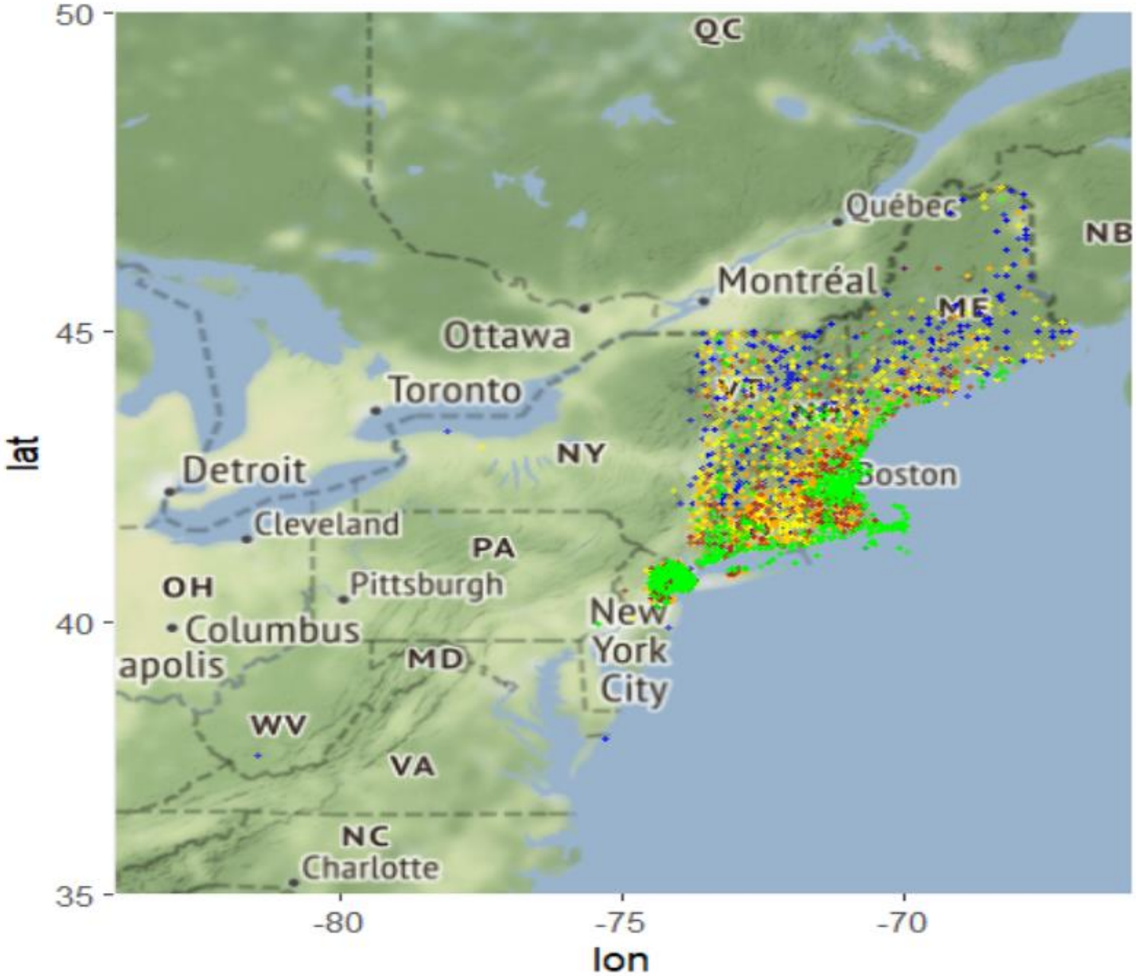
Note, the fit was better better for the decision tree—mean RSS of 191196646630—vs the random forest—236120844179--. Though not expected, this may have been due to the fact that the random forest only averaged over five trees, a fairly low number of trees which was selected to ensure that the algorithm would converge given the large number of observations (see limitations section).

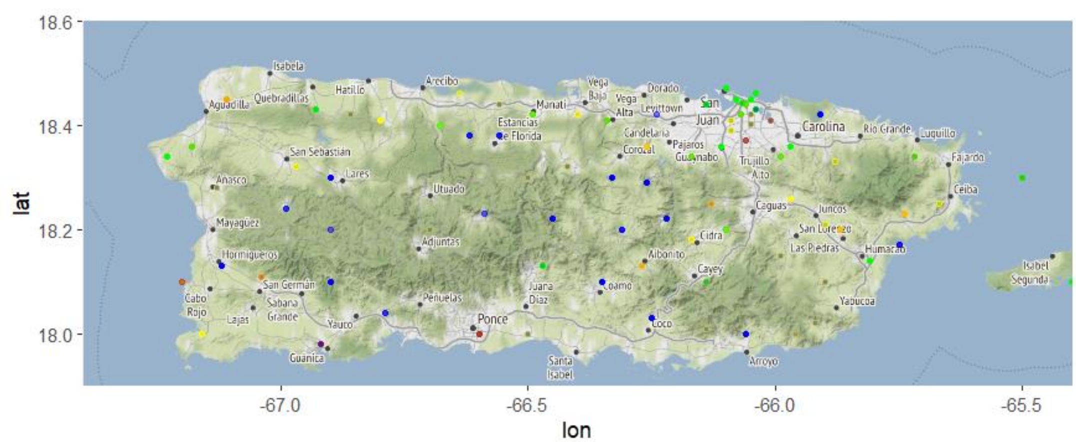
The other important aspect to note is the importance of state in the decision tree. States, particularly NY have the highest importance and factor into the first and next subsequent splits. As a result geographic information should be plotted.

7.1.6 Geographic Maps:

Given the importance of geography, each listing was plotted on the map to see if patterns could be detected in price. Latitude and longitude of each home listing were plotted, and to make the visualization more powerful, prices were binned in the following categories:

1. Price less than 250,000-blue
2. Price between 250,000 to 500,000-yellow
3. Price between 500,000 and 750,000-orange
4. Price between 750,000 and 1,000,000—brown
5. Price above 1,000,000—green





Here it becomes apparent that cities in the Boston-Washington corridor, along with Urban San Juan, had the highest prices. The map appears to confirm the findings in the decision tree.

7.2 Classification:

Along with predicting price, this report aims to predict whether a home will sell or not through classification. Using the literature (see section 2). For this section, using the suggested models, the following models were used:

Parametric

1. Logistic Regression

Non-Parametric

1. Decision Trees

Mixed Model

1. Geographic Mapping

As noted in the section 5, has_sold is the target of interest, with 0 being dummy variable has not sold, and 1 being dummy variable sold.

Note, authors tended to shy away from using GAM models in classification for real estate, so this has model, relative to price prediction, has been excluded.

7.2.1 Logistic Regression:

A logistic regression attempts to apply a linear model (similar to linear regression) to a classification problem by converting the probability of an event occurring—to an odds ratio—and then taking the log odds, thus moving from an initial range of [0,1] to a range of $(-\infty, \infty)$, as is required by a linear model (James et al, 2021, Chapter 4).

As such the logistic model takes a log odds to create a linear system:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X.$$

James et al., (2021, p. 135).

Below is the output from the logistic regression:

```
> log.fit <- glm(has_sold~price+bed+bath+acre_lot+house_size+breakeven+new_bu
ild+state_pr+state_ma+state_ct+state_nj+state_nh+ state_vt+state_ny+state_ri+
state_va+state_me+state_pa+state_wv, family = binomial, data = trainset2)
> summary(log.fit)

Call:
glm(formula = has_sold ~ price + bed + bath + acre_lot + house_size + 
    breakeven + new_build + state_pr + state_ma + state_ct + 
    state_nj + state_nh + state_vt + state_ny + state_ri + state_va + 
    state_me + state_pa + state_wv, family = binomial, data = trainset2)

Coefficients: (1 not defined because of singularities)
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -2.931e+10  1.425e+12  -0.021  0.98359    
price       -1.257e-07  1.859e-08  -6.760 1.38e-11 ***
bed          2.536e-02  9.803e-03   2.587  0.00968 ** 
bath         5.701e-02  1.425e-02   4.002 6.29e-05 ***
```

```
acre_lot    -4.204e-01  2.163e-02 -19.433  < 2e-16 ***
house_size  -3.209e-05  1.741e-05  -1.844  0.06524 .
breakeven   -3.581e-04  6.754e-04  -0.530  0.59597
new_build   -1.641e+01  3.528e+02  -0.047  0.96290
state_pr     2.931e+10  1.425e+12   0.021  0.98359
state_ma     2.931e+10  1.425e+12   0.021  0.98359
state_ct     2.931e+10  1.425e+12   0.021  0.98359
state_nj     2.931e+10  1.425e+12   0.021  0.98359
state_nh     2.931e+10  1.425e+12   0.021  0.98359
state_vt     2.931e+10  1.425e+12   0.021  0.98359
state_ny     2.931e+10  1.425e+12   0.021  0.98359
state_ri     2.931e+10  1.425e+12   0.021  0.98359
state_va     2.931e+10  1.425e+12   0.021  0.98359
state_me     2.931e+10  1.425e+12   0.021  0.98359
state_pa     2.931e+10  1.425e+12   0.021  0.98359
state_wv            NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 59760  on 43107  degrees of freedom
Residual deviance: 53158  on 43089  degrees of freedom
AIC: 53196

Number of Fisher Scoring iterations: 15
```

Similar to the linear regression, price, bed, bath, acre_lot were statistically significant. As such, the model was re run with new_build and states removed

```
> log.fit3 <- glm(has_sold~price+bed+bath+acre_lot+house_size+breakeven, fami
ly = binomial, data = trainset2)
> summary(log.fit3)

Call:
glm(formula = has_sold ~ price + bed + bath + acre_lot + house_size +
    breakeven, family = binomial, data = trainset2)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.348e-01  3.195e-02   4.218 2.47e-05 ***
price        2.151e-08  1.560e-08   1.379  0.16786
bed          6.344e-03  9.238e-03   0.687  0.49223
bath         9.761e-02  1.311e-02   7.446 9.61e-14 ***
acre_lot    -5.789e-01  1.931e-02 -29.985  < 2e-16 ***
house_size  -7.739e-05  1.612e-05  -4.801 1.58e-06 ***
breakeven    1.668e-03  5.124e-04   3.256  0.00113 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 59760  on 43107  degrees of freedom
Residual deviance: 58592  on 43101  degrees of freedom
AIC: 58606

Number of Fisher Scoring iterations: 4
```

Here, we can see a tighter fit, with almost all features except for price and bed being statistically significant.

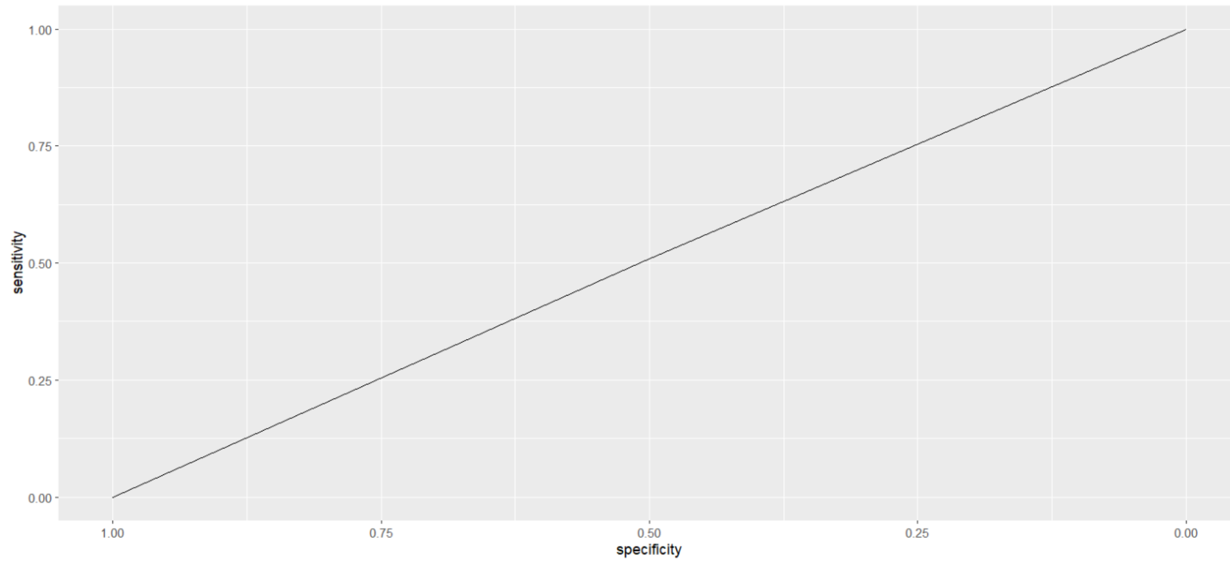
Next, to gain better insight, confidence intervals were generated:

```
> confidence
                    2.5 %        97.5 %
(Intercept)  7.360299e-02  1.988766e-01
price       -7.907944e-09  5.326449e-08
bed         -1.293377e-02  2.329107e-02
bath         6.933052e-02  1.207574e-01
acre_lot    -6.179833e-01 -5.423045e-01
house_size  -1.035421e-04 -4.025131e-05
breakeven    6.676171e-04  2.676525e-03
new_build              NA -9.258086e+00
```

And a confusion matrix for misclassification, along with a ROC curve plotting true positive to false positive rates were created:

```
> confmatrix
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2703 2744
         1 2690 2639

               Accuracy : 0.4957
                 95% CI : (0.4862, 0.5052)
    No Information Rate : 0.5005
    P-Value [Acc > NIR] : 0.8395

                  Kappa : -0.0085

 Mcnemar's Test P-Value : 0.4722

            Sensitivity : 0.5012
            Specificity : 0.4902
         Pos Pred Value : 0.4962
         Neg Pred Value : 0.4952
             Prevalence : 0.5005
         Detection Rate : 0.2508
   Detection Prevalence : 0.5055
      Balanced Accuracy : 0.4957

       'Positive' Class : 0
```

ROC Curve:



Based on the outputs, the accuracy was `0.4957 (.5043` missclassified), resulting in a straight line ROC curve.

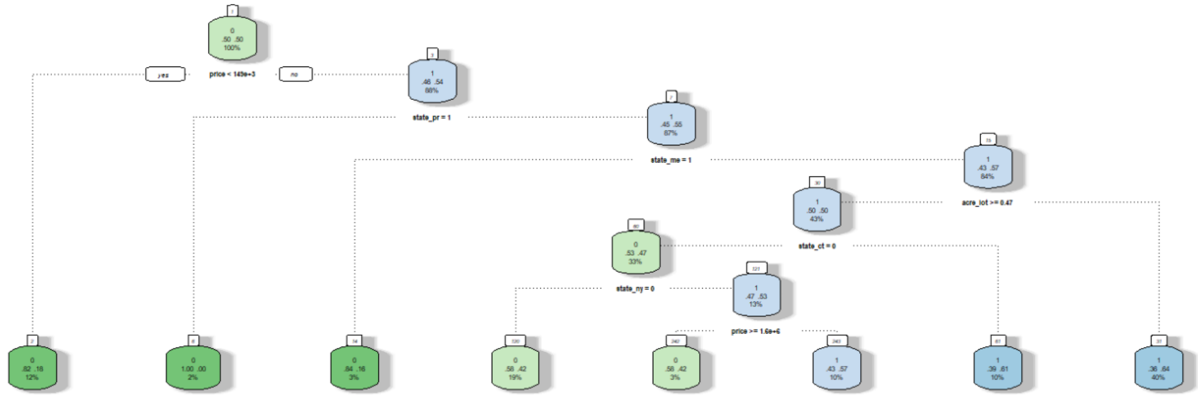
Given the poor fit, non-parametric models were next examined.

7.2.2 Trees: Classify Decision Tree and Classification Random Forest

Next, a decision trees and random forests were generated. The concept behind trees that classify is similar to that of the regression trees (shown earlier) but the regions correspond to a specific, non-overlapping regions, whose design is to minimize misclassification error, gini, and entropy (James, et al., 2021, p. 335-336). Note: while normally gini and entropy are used to test the accuracy of the model, this report uses misclassification so as to allow comparison to the logistic regression.

Below are the outputs from the decision tree including the predictions on test set data and the misclassification rate:

Call: treeclass_train <- rpart(has_sold ~ price+bed+bath+acre_lot+house_size+breakeven+new_build+ state_pr+state_ma+state_ct+state_nj+state_nh+state_vt+state_ny+state_ri+state_va+state_me +state_pa, data = trainset2, method = "class")



Confusion Matrix and Misclassification:

```
> conf_matrixtree
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2910 2483
         1 1345 4038

               Accuracy : 0.6448
                 95% CI : (0.6356, 0.6538)
    No Information Rate : 0.6051
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2897

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6839
            Specificity : 0.6192
         Pos Pred Value : 0.5396
         Neg Pred Value : 0.7501
             Prevalence : 0.3949
         Detection Rate : 0.2700
   Detection Prevalence : 0.5005
      Balanced Accuracy : 0.6516

       'Positive' Class : 0
```

From above, the accuracy rate climbed to 0.6448 (.3552 misclassified) compared to 0.4957 (.5043 misclassified) for the logistic regression. Clearly, the non parametric model performed better. However, to gain further insight, random forest was conducted.

```
> rf_classification <- randomForest(has_sold ~ price+bed+bath+acre_lot+house_
size+breakeven+new_build+state_pr+state_ma+state_ct+state_nj+state_nh+ state_
vt+state_ny+state_ri+state_va+state_me+state_pa, data = trainset2, mtry = 5,
ntree=5)
> rf_classification

Call:
 randomForest(formula = has_sold ~ price + bed + bath + acre_lot +      house
_size + breakeven + new_build + state_pr + state_ma +      state_ct + state_n
```

```
j + state_nh + state_vt + state_ny + state_ri +      state_va + state_me + st
ate_pa, data = trainset2, mtry = 5,      ntree = 5)
               Type of random forest: classification
                     Number of trees: 5
No. of variables tried at each split: 5

        OOB estimate of  error rate: 33.51%
Confusion matrix:
      0     1 class.error
0 11901  7531   0.3875566
1  5445 13841   0.2823292
```

And a confusion matrix for misclassification of test data (the above matrix is for trainset data) was run:

```
> conf_matrixrf
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3320 2073
         1 1269 4114

               Accuracy : 0.6899
                 95% CI : (0.681, 0.6986)
    No Information Rate : 0.5741
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3798

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.7235
            Specificity : 0.6649
         Pos Pred Value : 0.6156
         Neg Pred Value : 0.7643
             Prevalence : 0.4259
         Detection Rate : 0.3081
   Detection Prevalence : 0.5005
      Balanced Accuracy : 0.6942

       'Positive' Class : 0
```

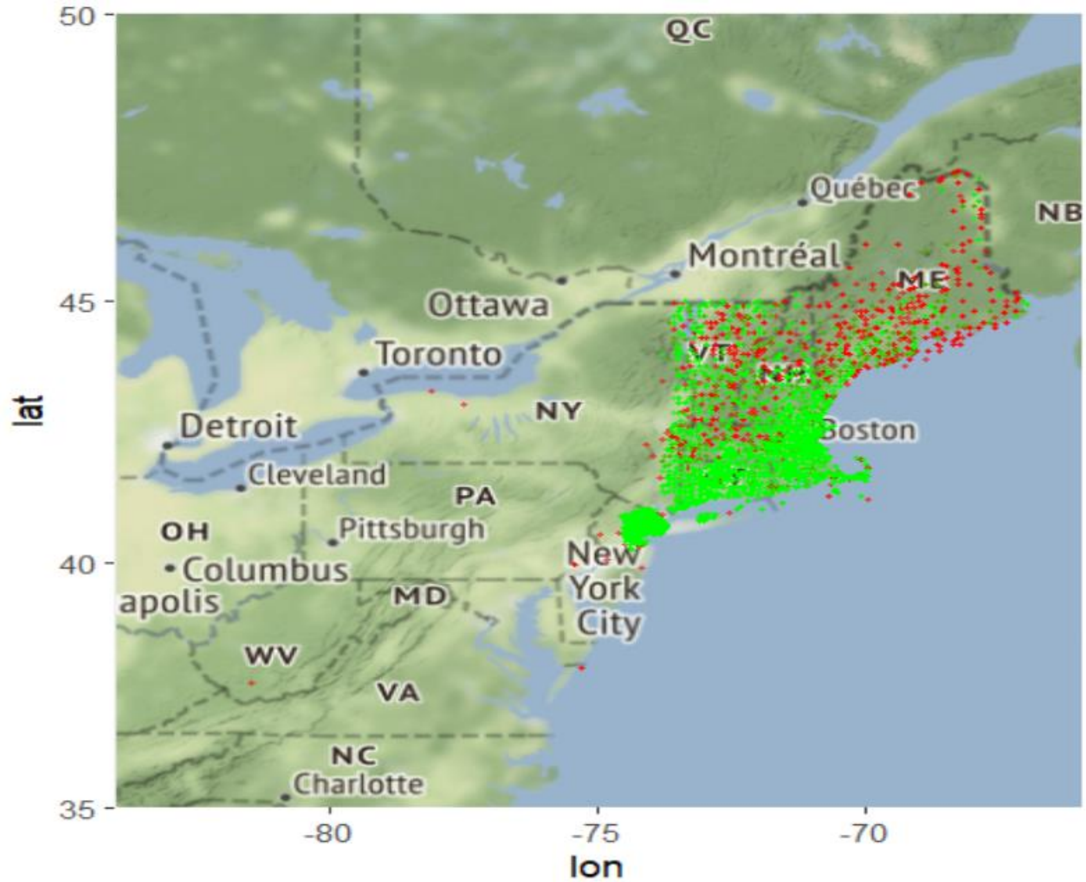
Here, an accuracy rate of 0.6899 (.3101 misclassification rate), was found relative to 0.6448 (.3552 misclassified) compared to 0.4957 (.5043 misclassified) the decision tree and logit models.

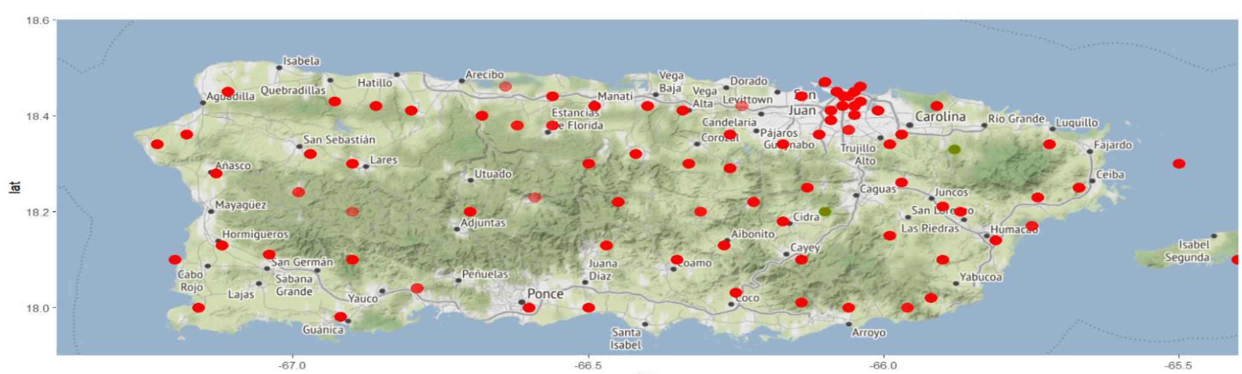
Again, looking at the tree, besides price, the most important split features appear to be states. As such, geocoordinate research was conducted.

7.2.3 Geographic mapping:

Maps were created plotting:

1. Non sold properties in red
2. Sold properties in green





From the outputs, it can be seen that homes in the Boston-Washington corridor have a greater chance of selling then those outside and in Puerto Rico.

Section 8 Limitations and Conclusion:

8.1 Conclusion:

Below are the accuracy readings—mean RSS for predicting price, and accuracy/misclassification for classification of sold—for each model (only the best subset used).

Model-Prediction	Accuracy-Mean RSS	Model Classification	Accuracy-Accuracy and Misclassification
Regression	302373223135	Logistic Regression	0.4957|.5043
Ridge Regression	302364263316	Decision Tree	0.6448 |.3552
Lasso	302361386135	Random Forest	0.6899|.3101
GAM	342084229665		
Decision Tree	191196646630		
Random Forest	236120844179		

All models—statistically significant at .05 level.

As can be seen above, for both prediction of price and classification of homes that sell, non-parametric tree methods tend to do a better job at accurately predicting and classifying. With this finding, looking the decision trees produced, location and price seem to have the greatest affect on whether or not a home will sell.

So to visualize the relationship between price and home sales, the following matrix was created:

```
> has_soldmatrix
            sold notsold
<250k       4221    8351
250k-500k  10160    7503
500k-750k   5407    4378
750k-1000k  2854    2410
1000k       4228    4372
```

Here, it becomes apparent that mid range price homes—250000 to 750000—tend to sell. Furthermore, those located in an urban area have a greater chance of selling. Furthermore, price itself is heavily based location, as evident based on the decision tree for price and cross checking the outputs against the price map. As such, given that homes located in urban areas tend to sell, and given that homes in these areas tend to be more expensive, those listed between 250000 and 750000 may be well priced in these regions and hence likely to sell.

8.2 Limitations:

As noted throughout this report, there are many limitations. In terms of modeling:

1. Conclusions—Misclassification. Normally misclassification is not used for evaluating decision trees and random forests. Misclassification was used her to ensure cross comparability with logistic regression.
2. Price was not normally distributed. Regression models assume independent variables are normally distributed, though a robust, and as such do not completely require it. Had there been more time, this variable should have been transformed into a normally distributed one, through bucketing.
3. The random forest for predicting price averaged over only 5 trees. Normally an accurate random forest should iterate over many more than 5, as each tree comprises of only n number of features,

which is calculated by taking the square root of the total number of features, in this case 18. Therefore, each simulation ran on $\sqrt{18}$ features. In order to accurate prediction, around 25 should be run (James et al 2021 Chapter 9). 5 was selected in this case to ensure that the model converged. In the future, a greater number of trees should be generated, and if needed, done on an external server

4. West Virginia and Virginia. These states had few to no observations, and hence effected the quality of the models. However, taking them out caused the model to perform worse, in general. Perhaps this indicates that location is very important. Future research should collect more data from these states to add to the research. Furthermore, regression and logistical regression best fit required removing states (not statistically significant). However, given literature indicated the importance of location, states were included in trees.
5. Quality of Data. The main data set used, USA Real Estate Data Set (Zillow) was created via web scrapping and placed on Kaggle. However, while conducting the web scrape, the user who uploaded this set to Kaggle accidently copied the same data over and over. This affected the quality of the data set.
6. Bias Variance Trade-Off. The models used were not calibrated in any way (such as adding non linear terms to a regression or taking square roots of the dependent variable, etc). This was done to avoid raising the bias too much, making models fit too closely to training data, while also making them uninterpretable to the nature of real estate. For example squaring or taking square roots of bed rooms or acre_lot may not be interpretable. Future research may warrant manipulating variables.
7. Heterogeneous Nature of Real Estate. Real estate tends to be heterogeneous. As such, it may be that real estate is in fact not very suitable at all for statistical and machine learning.

References:

Data Sets:

1. USA Real Estate Data Set (Zillow), from Kaggle. Available at: [USA Real Estate Dataset | Kaggle]
2. Buy vs Rent, (Zillow) from Kaggle, Available at: [Buy vs Rent - dataset by zillow-data | data.world]
3. Zip-Code-To-County, From GitHub, Available at: [zip-code-to-county/county-fips.csv at master · Data4Democracy/zip-code-to-county · GitHub]
4. US Zipcode to County State to FIPS lookup, from Data.world: Available at: [niccolley/us-zipcode-to-county-state | Workspace | data.world]
5. States.csv file, a file the group made with state names and their two letter shortened abbreviation
6. Fips2County.tsv, from GitHub. Available at: [articles/fips2county.tsv at master · ChuckConnell/articles · GitHub]

Libraries Used:

Wickham H, Vaughan D, Girlich M (2023). _tidyr: Tidy Messy Data_. R package version 1.3.0, <https://CRAN.R-project.org/package=tidyr>.

G.C. Rozzi, zipcodeR: Advancing the analysis of spatial data at the ZIP code level in R, Softw. Impacts. (2021) 100099.

Grothendieck G (2017). _sqldf: Manipulate R Data Frames Using SQL_. R package version 0.4-11, <https://CRAN.R-project.org/package=sqldf>.

Wickham H, François R, Henry L, Müller K, Vaughan D (2023). _dplyr: A Grammar of Data Manipulation_. R package version 1.1.2, <https://CRAN.R-project.org/package=dplyr>.

Wickham H (2022). _stringr: Simple, Consistent Wrappers for Common String Operations_. R package version 1.5.0, <https://CRAN.R-project.org/package=stringr>.

Gross J, Ligges U (2015). _nortest: Tests for Normality_. R package version 1.0-4, <https://CRAN.R-project.org/package=nortest>.

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. Journal of Statistical Software, 28(5), 1–26. https://doi.org/10.18637/jss.v028.i05

Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti, Frédérique Lisacek, Jean-Charles Sanchez and Markus Müller (2011). pROC: an open-source package for R and S+ to analyze and compare ROC curves. BMC Bioinformatics, 12, p. 77. DOI: 10.1186/1471-2105-12-77 <http://www.biomedcentral.com/1471-2105/12/77/>

Harrell Jr F (2023). _Hmisc: Harrell Miscellaneous_. R package version 5.1-0, <https://CRAN.R-project.org/package=Hmisc>.

Friedman J, Tibshirani R, Hastie T (2010). “Regularization Paths for Generalized Linear Models via Coordinate Descent.” _Journal of Statistical Software_, *33*(1), 1-22. doi:10.18637/jss.v033.i01 <https://doi.org/10.18637/jss.v033.i01>.

Miller TLboFcbA (2020). _leaps: Regression Subset Selection_. R package version 3.1, <https://CRAN.R-project.org/package=leaps>.

Wood, S.N. (2011) Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. Journal of the Royal Statistical Society (B) 73(1):3-36

Wood S.N., N. Pya and B. Saefken (2016) Smoothing parameter and model selection for general smooth models (with discussion). Journal of the American Statistical Association 111:1548-1575.

Wood, S.N. (2004) Stable and efficient multiple smoothing parameter estimation for generalized additive models. Journal of the American Statistical Association. 99:673-686.

Wood, S.N. (2017) Generalized Additive Models: An Introduction with R (2nd edition). Chapman and Hall/CRC.

Wood, S.N. (2003) Thin-plate regression splines. Journal of the Royal Statistical Society (B) 65(1):95-114.

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

Therneau T, Atkinson B (2022). _rpart: Recursive Partitioning and Regression Trees_. R package version 4.1.19, <https://CRAN.R-project.org/package=rpart>.

Milborrow S (2022). _rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'_. R package version 3.1.1, <https://CRAN.R-project.org/package=rpart.plot>.

Williams, G. J. (2011), Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery, Use R!, Springer.

Neuwirth E (2022). _RColorBrewer: ColorBrewer Palettes_. R package version 1.1-3, <https://CRAN.R-project.org/package=RColorBrewer>.

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

Hothorn T, Hornik K, Zeileis A (2006). “Unbiased Recursive Partitioning: A Conditional Inference Framework.” _Journal of Computational and Graphical Statistics_, *15*(3), 651-674. doi:10.1198/106186006X133933 <https://doi.org/10.1198/106186006X133933>.

Zeileis A, Hothorn T, Hornik K (2008). “Model-Based Recursive Partitioning.” _Journal of Computational and Graphical Statistics_, *17*(2), 492-514. doi:10.1198/106186008X319331 <https://doi.org/10.1198/106186008X319331>.

Hothorn T, Buehlmann P, Dudoit S, Molinaro A, Van Der Laan M (2006). “Survival Ensembles.” _Biostatistics_, *7*(3), 355-373.

Strobl C, Boulesteix A, Zeileis A, Hothorn T (2007). “Bias in Random Forest Variable Importance Measures: Illustrations, Sources and a Solution.” _BMC Bioinformatics_, *8*(25). doi:10.1186/1471-2105-8-25 <https://doi.org/10.1186/1471-2105-8-25>.

Strobl C, Boulesteix A, Kneib T, Augustin T, Zeileis A (2008). “Conditional Variable Importance for Random Forests.” _BMC Bioinformatics_, *9*(307). doi:10.1186/1471-2105-9-307 <https://doi.org/10.1186/1471-2105-9-307>.

Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, Chen K, Mitchell R, Cano I, Zhou T, Li M, Xie J, Lin M, Geng Y, Li Y, Yuan J (2023). _xgboost: Extreme Gradient Boosting_. R package version 1.7.5.1, <https://CRAN.R-project.org/package=xgboost>.

Wickham H, Hester J, Bryan J (2023). _readr: Read Rectangular Text Data_. R package version 2.1.4, <https://CRAN.R-project.org/package=readr>.

Fox J, Weisberg S (2019). _An R Companion to Applied Regression_, Third edition. Sage, Thousand Oaks CA. <https://socialsciences.mcmaster.ca/jfox/Books/Companion/>.

Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2023). _e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien_. R package version 1.7-13, <https://CRAN.R-project.org/package=e1071>.

Peters A, Hothorn T (2023). _ipred: Improved Predictors_. R package version 0.9-14, <https://CRAN.R-project.org/package=ipred>.

D. Kahle and H. Wickham. ggmap: Spatial Visualization with ggplot2. The R Journal, 5(1), 144-161. URL http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf

Library (CRAN)	Documentation
tidyr	tidyr package - RDocumentation
zipcodeR	zipcodeR package - RDocumentation
sqldf	sqldf package - RDocumentation
dplyr	dplyr package - RDocumentation
stringr	stringr package - RDocumentation
nortest	nortest package - RDocumentation
caret	caret package - RDocumentation
pROC	pROC package - RDocumentation
Hmisc	Hmisc package - RDocumentation
glmnet	glmnet package - RDocumentation
leaps	glmnet package - RDocumentation
mgcv	mgcv package - RDocumentation
ggplot2	ggplot2 package - RDocumentation
rpart	rpart package - RDocumentation
rpart.plot	rpart.plot package - RDocumentation
rattle	rattle package - RDocumentation
RColorBrewer	RColorBrewer package - RDocumentation
randomForest	randomForest package - RDocumentation
party	party package - RDocumentation
xgboost	xgboost package - RDocumentation
readr	readr package - RDocumentation
car	car package - RDocumentation
e1071	e1071 package - RDocumentation
ipred	ipred package - RDocumentation
ggmap	ggmap package - RDocumentation
ggmap (github installation)	ggmap package - RDocumentation

Literature:

1. Baldominos, A, Blanco, I, Moreno, AJ, Iturrarte, R, Bernardez, O, Afonso, C (2018), "identifying real estate opportunities using machine learning", Applied Science 2018(8), pp. 1-24
2. Calainho, FD, de Minne, A M V, Francke, M K (2022), "a machine learning approach to price indices: applications in commercial real estate", Journal of Real Estate Finance and Economics.
3. Chaillou, S, Fink, D, Goncalves, P (2017), "urban tech on the rise: machine learning disrupts the real estate industry", FACTS Reports 2017(17).
4. Forys, I (2022), "machine learning in house price analysis: regression models versus neural networks", Procedia Computer Science 207, pp. 435-445.
5. Grybauskas, A, Pilinkiene, V, Stundziene, A (2021), "predictive analytics using big data for the real estate market during the COVID-19 pandemic" J of Big Data, pp. 1-20.
6. Lorenz, F, Willwersch, J, Cajias, M, Fuerst, F (2022), "interpretable machine learning for real estate market analysis", Real Estate Economics
7. Trawinski, B, Telec, Z, Piwowarczyk, M, Lasota T (2017), "comparison of expert algorithms with machine learning models for real estate appraisal", Conference Paper via Research Gate.
8. Xiao, Y (2022), "big data for comprehensive analysis of real estate market", California State University San Bernardino.
9. James G, Witten, D, Hastie, T, Tibshirani, R (2021), Introduction to Statistical Learning, 2nd Edition
10. Hu, H., Wu, J (2016), "Real Estate Price Prediction with Regression and Classification, Stanford University, Available at: WuYu_HousingPrice_report.pdf (stanford.edu).
11. Bailey, JR, Lauria, D, Lindquist, WB, Mittnik, S, Rachev, ST (2022), "Hedonic models of real estate prices: GAM Models; Environment, and Sex-Offender-Proximity-Factors", Journal of Risk and Financial Management 15(601).
12. James, G, Witten, D, Hastie, T, Tibshirani, R (2021), Introduction to Statistical Learning With Applications in R, 2nd Edition, Online.