Predicting Home Value Based on Structural Features

Maria Mercier

June 23. 2018

Introduction

Purchasing a home is a major milestone in life. Individuals spend an enormous amount of time researching the various characteristics of a house to find the best value. One helpful resource would be to predict the value of homes based on the structural features and then add other characteristics to that for comparison purposes. This project was developed to examine such a concept.

There were several objectives I wanted to acheive during this course. First, I wanted to develop a predictive model that could potentially be applied in a similar geographical areas. I also wanted to explore various regression methods and functions. Another goal was to identify which features were most important to the model. Therefore, I formulated a class project to apply a linear regression machine learning techniques to predict the value of a home based on its structural features.

The project consisted of using data from the Kaggle website titled Zillow "properties_2016". The project is limited to structural features so that this model may be applicable in other similar geographical areas (with additional data from datasets in different geographical areas).

The prepatory phase for developing the prediction model included importing data, exploration of the data, selecting the features, and data cleansing. Regression models were created using various regression functions in R and compared. This paper decribes each step in the process along with the code and output. You may view a twelve and a half minute video summarizint this project on YouTube at https://www.youtube.com/watch?v=-yyfe9LdPDc&feature=youtu.be.

Dataframe

The real estate data set is the 2016 Zillow properties and was taken from the Kaggle web site. This data set consisted of 3 million observations on 58 features from homes in the Los Angeles area. The Zillow data set contained features on various information about a home, such as living area, types of rooms, the presence of a fire place, number of garages, the presence of a pool etc. It also contained geographical information such as longitude and latitude, land codes, zoning, and region of the city. The dataset properties_2016.csv.zip was obtained from https://www.kaggle.com/c/zillow-prize-1/data.

Exploratory Data Analysis and Data Cleansing

The exploratory data analysis included examining the data frame for feature definitions, missing values, duplicate features and extreme values. This data set containing all of these issues. The first step was to view the structure of the dataframe and is shown below.

```
## 'data.frame':    2985217 obs. of  58 variables:
##  $ parcelid                    : int  10754147 10759547 10843547 10859147 10879947 10898347 10933547
##  $ airconditioningtypeid       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ architecturalstyletypeid    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ basementsqft                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ bathroomcnt                 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ bedroomcnt                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ buildingclasstypeid         : int  NA NA NA 3 4 4 NA NA NA 3 ...
```

```
##  $ buildingqualitytypeid       : int  NA NA NA 7 NA 7 NA NA NA 7 ...
##  $ calculatedbathnbr           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ decktypeid                  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ finishedfloor1squarefeet    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ calculatedfinishedsquarefeet: num  NA NA 73026 5068 1776 ...
##  $ finishedsquarefeet12        : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ finishedsquarefeet13        : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ finishedsquarefeet15        : int  NA NA 73026 5068 1776 2400 NA 3611 NA 3754 ...
##  $ finishedsquarefeet50        : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ finishedsquarefeet6         : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ fips                        : int  6037 6037 6037 6037 6037 6037 6037 6037 6037 6037 ...
##  $ fireplacecnt                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ fullbathcnt                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ garagecarcnt                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ garagetotalsqft             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ hashottuborspa              : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ heatingorsystemtypeid       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ latitude                    : int  34144442 34140430 33989359 34148863 34194168 34171873 34131929
##  $ longitude                   : int  -118654084 -118625364 -118394633 -118437206 -118385816 -1183809
##  $ lotsizesquarefeet           : num  85768 4083 63085 7521 8512 ...
##  $ poolcnt                     : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ poolsizesum                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ pooltypeid10                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ pooltypeid2                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ pooltypeid7                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ propertycountylandusecode   : Factor w/ 241 levels "","0","010","0100",..: 14 12 152 152 164 164 2
##  $ propertylandusetypeid       : int  269 261 47 47 31 31 260 31 269 31 ...
##  $ propertyzoningdesc          : Factor w/ 5639 levels "","#12","**AHRP",..: 1 2159 1817 1817 1838 18
##  $ rawcensustractandblock      : num  60378002 60378001 60377030 60371412 60371232 ...
##  $ regionidcity                : int  37688 37688 51617 12447 12447 12447 12447 396054 396054 47547
##  $ regionidcounty              : int  3101 3101 3101 3101 3101 3101 3101 3101 3101 3101 ...
##  $ regionidneighborhood        : int  NA NA NA 27080 46795 46795 274049 NA NA NA ...
##  $ regionidzip                 : int  96337 96337 96095 96424 96450 96446 96049 96434 96436 96366 ..
##  $ roomcnt                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ storytypeid                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ threequarterbathnbr         : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ typeconstructiontypeid      : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ unitcnt                     : int  NA NA 2 NA 1 NA NA NA NA NA ...
##  $ yardbuildingsqft17          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ yardbuildingsqft26          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ yearbuilt                   : num  NA NA NA 1948 1947 ...
##  $ numberofstories             : int  NA NA NA 1 NA 1 NA 1 NA 1 ...
##  $ fireplaceflag               : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ structuretaxvaluedollarcnt  : num  NA NA 650756 571346 193796 ...
##  $ taxvaluedollarcnt           : num  9 27516 1413387 1156834 433491 ...
##  $ assessmentyear              : int  2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 ...
##  $ landtaxvaluedollarcnt       : num  9 27516 762631 585488 239695 ...
##  $ taxamount                   : num  NA NA 20800 14558 5725 ...
##  $ taxdelinquencyflag          : Factor w/ 2 levels "","Y": 1 1 1 1 1 1 1 1 1 1 ...
##  $ taxdelinquencyyear          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ censustractandblock         : num  NA NA NA NA NA NA NA NA NA NA ...
```

The goal of this project is to predict the value of a home from its structural features. The response variable is the column "structuretaxvaluedollarcnt"" in the original dataframe, which is continuous type data. The machine learning model will be a multiple linear regression model as recommended by Lantz (2015 p. 21). The next step was to create a new variable for dataframe to preserve original dataset.

```
properties <- properties_2016
```

Next, I changed column names to more sensical names as done by Spachtholz (2017) found at https://www.kaggle.com/philippsp/exploratory-analysis-zillowin Kaggle except used building_value label for predictor.

One component of the exploratory data analysis was to check for duplicate features and values. There were two columns that appeared similar and they are: area_total_calc and area_total_finished. This was verified by looking at the head and tail of those columns. The feature area_total_calc has fewer missing values so I will remove area_total_finished later.

I also checked the response variable building_value for missing values, to see how many there were.

```
sum(is.na(properties$building_value))
```

```
## [1] 54982
```

There were 54,982 observations with missing vaues. I decided to delete the rows with missing values for response feature as they are only 2% of the rows in the data set. This code is from Clemens (2018). Then I created a new dataset without missing values.

I verifed that the rows with NA in this column were removed.

```
sum(is.na(properties_no_building_na$building_value))
```

```
## [1] 0
```

The next step was to check percent of missing values for all features and the code for this step was taken from Walters (2017).

The goal for this project is to predict the value of the building based on its structural features. I had to remove obvious features that are not related to the structure of the home. This was determined based on the definitions of each feature provided in an excel spreadsheet. The spread sheet with the definitions for each variable was provided on the Zillow website and was taken from https://www.kaggle.com/philippsp/exploratory-analysis-zillow/data,

To remove the non-structural features (including building_year) I used the code from https://stackoverflow.com/questions/5234117/how-to-drop-columns-by-name-in-a-data-frame. I also removed the feature area_live_finished which appears to be a duplicate column of area_total_calc.

```
struct_features_only_data <- within(properties_no_building_na, rm('fips', 'latitude', 'longitude', 'area
```

The new dataset struture is listed below.

```
## 'data.frame':    2930235 obs. of  30 variables:
##  $ aircon                  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ architectural_style     : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_basement           : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_bathroom            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num_bedroom             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ framing                 : int  NA 3 4 4 NA NA 3 4 4 NA ...
##  $ quality                 : int  NA 7 NA 7 NA NA 7 NA 7 7 ...
##  $ num_bathroom_calc       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ deck                    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_firstfloor_finished: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_total_calc         : num  73026 5068 1776 2400 NA ...
##  $ area_liveperi_finished  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_total_finished     : int  73026 5068 1776 2400 NA 3611 3754 2470 2760 NA ...
```

```
##  $ area_unknown              : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_base                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_fireplace             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_bath                  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_garage                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ area_garage               : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ flag_tub                  : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ heating                   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_room                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ story                     : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_75_bath               : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ material                  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_unit                  : int  2 NA 1 NA NA NA NA NA NA NA ...
##  $ area_patio                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_story                 : int  NA 1 NA 1 NA 1 1 1 NA NA ...
##  $ flag_fireplace            : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ building_value            : num  650756 571346 193796 176383 397945 ...
```

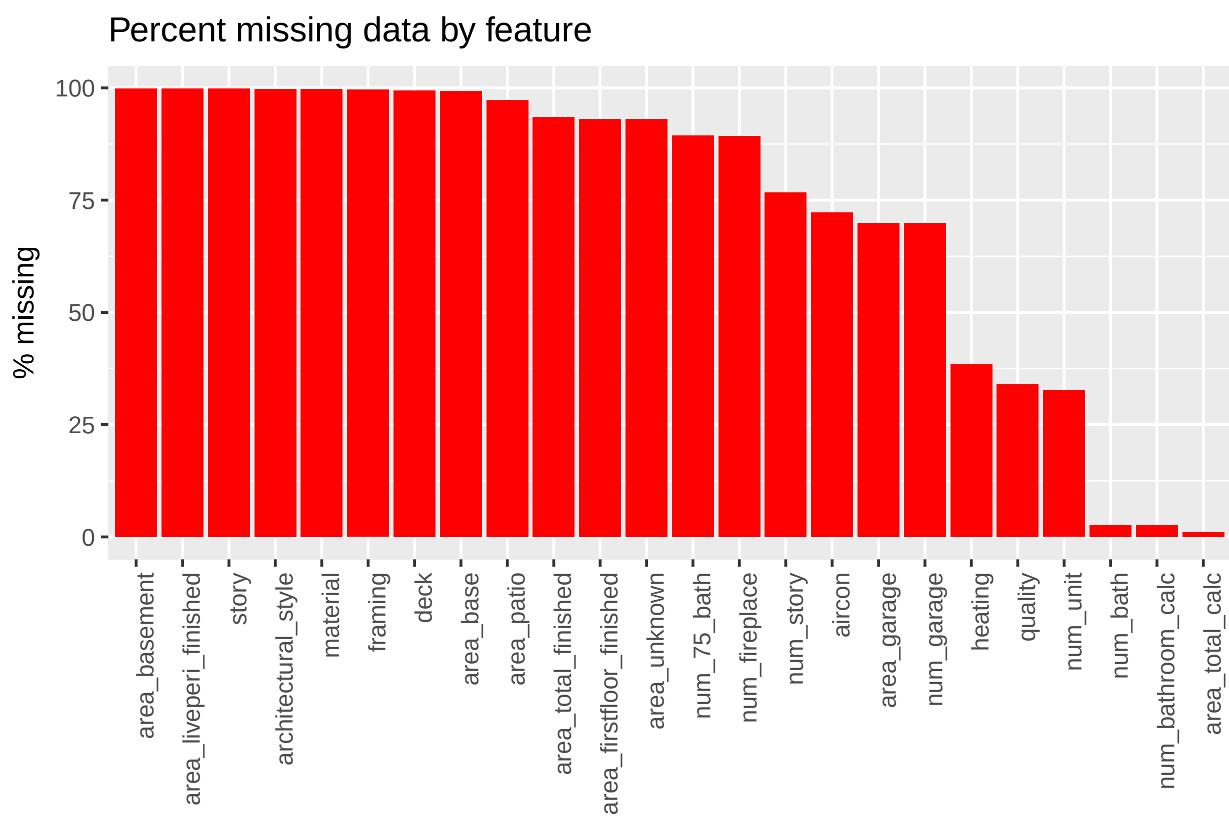
I checked to see if there any complete cases in the new dataset and if so, the number of them to bypass extensive data cleansing.

```
sum(complete.cases(struct_features_only_data))
```

```
## [1] 0
```

No complete cases, I proceeded with exploratory data anaysis (EDA) and data cleansing.

The next step was to examine the amount of missing data in the new working data set with structural features



only.

```
miss_pct
```

```
##                  aircon      architectural_style            area_basement
##                    72.3                     99.8                     99.9
##                 framing                  quality        num_bathroom_calc
##                    99.6                     34.0                      2.6
##                    deck area_firstfloor_finished          area_total_calc
##                    99.4                     93.1                      1.1
##   area_liveperi_finished      area_total_finished             area_unknown
##                    99.9                     93.5                     93.1
##               area_base             num_fireplace                 num_bath
##                    99.3                     89.3                      2.6
##              num_garage              area_garage                  heating
##                    69.9                     69.9                     38.4
##                   story               num_75_bath                 material
##                    99.9                     89.4                     99.8
##                num_unit               area_patio                num_story
##                    32.6                     97.3                     76.7
```

The next step was to select the features I wanted to include in the model. I wanted this model to have the potential to be used for any similar geographical areas, so I focused on the structural features of the building, and excluded geographical and neighborhood features. Also excluded were any structural features which contained 26% or more NAs. This residual sample set contained 6 features. The response variable is the building value. The predictors are the number of bathrooms, the total finished living area, the number of bedrooms, the presence of a fireplace and the presence of a hot tub or spa. The structure and summary of the dataset are shown below.

```
## 'data.frame':    2930235 obs. of  6 variables:
##  $ num_bathroom   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ area_total_calc: num  73026 5068 1776 2400 NA ...
##  $ num_bedroom    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ flag_fireplace : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ flag_tub       : Factor w/ 2 levels "","true": 1 1 1 1 1 1 1 1 1 1 ...
##  $ building_value : num  650756 571346 193796 176383 397945 ...
```

In order to run the regression model on this dataframe, all the features must be numeric. Therefore, I had to change flag_fireplace and flag_tub to binary features for the correlation matrix and the regression model. The original data set listed the presence of a fire place, hot tub or spa as "true". I change these character features into a numeric feature where the number one indicated that a fireplace, hot tube or spa were present in the home and zero to indicate otherwise. Since the homes in this data set are located in Los Angeles, its not surprising that there were few homes with these features.

```
##
##       0       1
## 2925072    5163
```

```
##
##       0       1
## 2861235   69000
```

Verifying the change to numeric.

```
##   num_bathroom area_total_calc num_bedroom flag_fireplace flag_tub
## 1            0           73026           0              0        0
## 2            0            5068           0              0        0
## 3            0            1776           0              0        0
## 4            0            2400           0              0        0
```

```
## 5           0            NA          0             0        0
## 6           0          3611          0             0        0
##   building_value
## 1         650756
## 2         571346
## 3         193796
## 4         176383
## 5         397945
## 6         101998
```

I then checked the class type of each variable in preparation for regression modeling.

```
##    num_bathroom area_total_calc     num_bedroom  flag_fireplace
##       "numeric"       "numeric"       "numeric"       "numeric"
##        flag_tub  building_value
##       "numeric"       "numeric"
```

Correlations

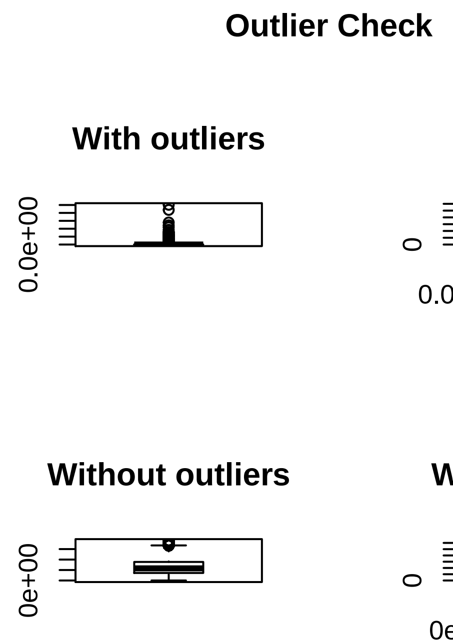
I ran a correlation matrix on the first dataset to see which features had a mild correlation with the response variable "building value". This correlation shows that three predictors had at least a weak to moderate correlation. They are the number of bedrooms, the number of bathrooms and the area_total_calc. According to Lantz (2015, p. 180) in the Chapter on Regression Methods, a weak correlation are values between 0.1 and 0.3, where as a moderate correlation is between 0.3 and 0.5. Strong correlations are values above 0.5. Despite these recommendations, I used all 5 predictors for the linear model to see how this played out. The code was taken from RDocumentation at https://www.rdocumentation.org/packages/stats/versions/3.5.0/topics/cor.

```
##                 num_bathroom area_total_calc num_bedroom flag_fireplace
## num_bathroom     1.000000000      0.35339107  0.68012096   -0.005089341
## area_total_calc  0.353391072      1.00000000  0.26186610   -0.009096870
## num_bedroom      0.680120957      0.26186610  1.00000000   -0.018949697
## flag_fireplace  -0.005089341     -0.00909687 -0.01894970    1.000000000
## flag_tub         0.127130815      0.08404231  0.07647492   -0.003197091
## building_value   0.273515348      0.62947348  0.12792031   -0.006405979
##                     flag_tub building_value
## num_bathroom     0.127130815    0.273515348
## area_total_calc  0.084042306    0.629473481
## num_bedroom      0.076474921    0.127920306
## flag_fireplace  -0.003197091   -0.006405979
## flag_tub         1.000000000    0.077746744
## building_value   0.077746744    1.000000000
```

Outliers

The next step was to remove the outliers within each feature. During this phase I encountered a bug in R studio. As I was removing the outliers for each feature, new NAs were added to other features in the form of new rows. I could see this in the summary of the data set. I tried the suggested remedies found on a Stackoverflow post at Subsetting R dataframe results in mysterious NA rows (n.d), such as using the filter(), using the subset() and recounting the rows but was unsuccessful.

I decided to use an outlier function I found on the internet and planned to removed all NAs since I had a large number of observations. I used a function from Dhana (2016) to remove outliers. This function provideds the mean, bloxplots and histogram before and after the outliers are remove. It will seek a response to the question “Do you want to replace outliers with NAs”. Once this is answered in the console it completes the function.

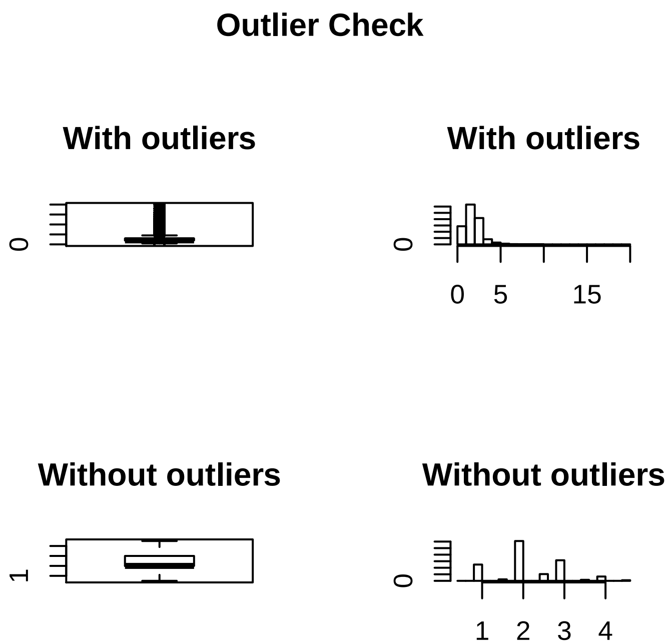


The outlier function applied to the building value and displayed in the graphs below

```
## Outliers identified: 190603 nPropotion (%) of outliers: 7 nMean of the outliers: 725608.3 nMean witho
## Nothing changed n
```

The above plots show the boxplots and histogram before and after the outliers were removed. The histogram after the building_value were removed is skewed to the right.

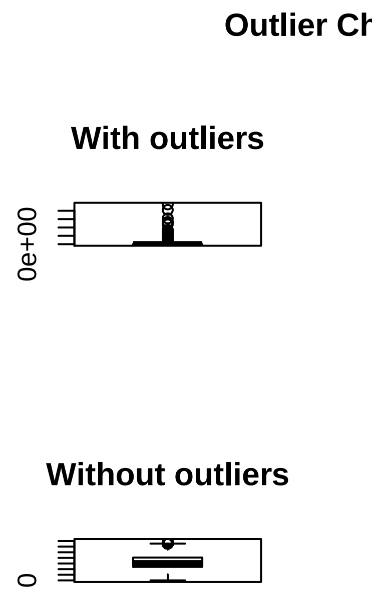
The outliers are remvoed from the num_bathroom features and shown in the graphs below




```
## Outliers identified: 151627 nPropotion (%) of outliers: 5.5 nMean of the outliers: 2.88 nMean withou
## Nothing changed n
```

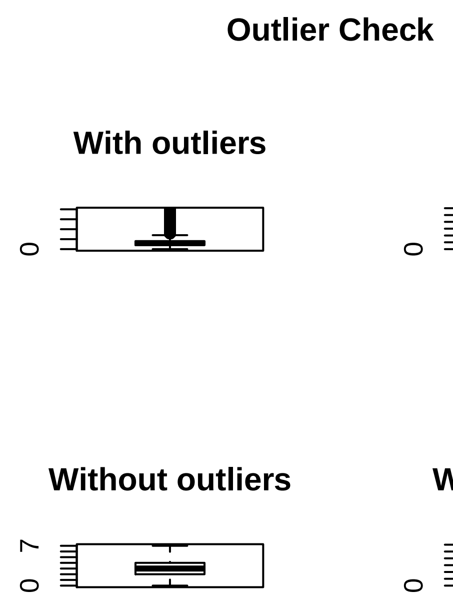
Check the summary to confirm outliers are remoed.

```
##   num_bathroom    area_total_calc   num_bedroom     flag_fireplace
##  Min.   : 0.000   Min.   :      1   Min.   : 0.000   Min.   :0.000000
##  1st Qu.: 2.000   1st Qu.:   1217   1st Qu.: 2.000   1st Qu.:0.000000
##  Median : 2.000   Median :   1579   Median : 3.000   Median :0.000000
##  Mean   : 2.239   Mean   :   1834   Mean   : 3.131   Mean   :0.001762
##  3rd Qu.: 3.000   3rd Qu.:   2145   3rd Qu.: 4.000   3rd Qu.:0.000000
##  Max.   :20.000   Max.   :952576   Max.   :20.000   Max.   :1.000000
##  NA's   :23       NA's   :33574    NA's   :12
##     flag_tub        building_value
##  Min.   :0.00000   Min.   :        1
##  1st Qu.:0.00000   1st Qu.:    74800
##  Median :0.00000   Median :   122590
##  Mean   :0.02355   Mean   :   170884
##  3rd Qu.:0.00000   3rd Qu.:   196889
##  Max.   :1.00000   Max.   :251486000
##
```



Removing outliers from area_total_calc with function and are shown in the graphs below.

```
## Outliers identified: 144335 nPropotion (%) of outliers: 5.2 nMean of the outliers: 5049.85 nMean wit
## Nothing changed n
```



Remove outliers from num_bedroom with function and shown in the graphs below.

```
## Outliers identified: 21154 nPropotion (%) of outliers: 0.7 nMean of the outliers: 8.7 nMean without
## Nothing changed n
```

Check the summary after all outliers were removed.

```
##   num_bathroom    area_total_calc   num_bedroom     flag_fireplace
##  Min.   : 0.000   Min.   :      1   Min.   : 0.000   Min.   :0.000000
##  1st Qu.: 2.000   1st Qu.:   1217   1st Qu.: 2.000   1st Qu.:0.000000
##  Median : 2.000   Median :   1579   Median : 3.000   Median :0.000000
##  Mean   : 2.239   Mean   :   1834   Mean   : 3.131   Mean   :0.001762
##  3rd Qu.: 3.000   3rd Qu.:   2145   3rd Qu.: 4.000   3rd Qu.:0.000000
##  Max.   :20.000   Max.   :952576   Max.   :20.000   Max.   :1.000000
##  NA's   :23       NA's   :33574    NA's   :12
##     flag_tub       building_value
##  Min.   :0.00000   Min.   :        1
##  1st Qu.:0.00000   1st Qu.:    74800
##  Median :0.00000   Median :   122590
##  Mean   :0.02355   Mean   :   170884
##  3rd Qu.:0.00000   3rd Qu.:   196889
##  Max.   :1.00000   Max.   :251486000
##
```

The next step was to identify the number of complete cases after outliers have been removed from all features in data25percent dataframe.

```
sum(complete.cases(data25percent))
```

```
## [1] 2896645
```

Then a new dataframe with complete cases was created.

Confirm the structure of the latest dataset.

```
## 'data.frame':    2896645 obs. of  6 variables:
##  $ num_bathroom   : num  0 0 0 0 0 0 0 0 0 2 ...
##  $ area_total_calc: num  73026 5068 1776 2400 3611 ...
##  $ num_bedroom    : num  0 0 0 0 0 0 0 0 0 4 ...
##  $ flag_fireplace : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ flag_tub       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ building_value : num  650756 571346 193796 176383 101998 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:33590] 5 27 31 34 35 36 47 54 65 67 ...
##   .. ..- attr(*, "names")= chr [1:33590] "5" "27" "31" "34" ...
```

Recheck correlation with clean dataframe complete_data25percent

```
##                 num_bathroom area_total_calc num_bedroom flag_fireplace
## num_bathroom     1.000000000     0.353391072  0.65963945   -0.006285553
## area_total_calc  0.353391072     1.000000000  0.26185108   -0.009096834
## num_bedroom      0.659639445     0.261851076  1.00000000   -0.020978611
## flag_fireplace  -0.006285553    -0.009096834 -0.02097861    1.000000000
## flag_tub         0.126706123     0.084045182  0.07479690   -0.003273021
## building_value   0.325193872     0.629475728  0.15449555   -0.007381048
##                     flag_tub building_value
## num_bathroom     0.126706123    0.325193872
## area_total_calc  0.084045182    0.629475728
## num_bedroom      0.074796899    0.154495554
## flag_fireplace  -0.003273021   -0.007381048
## flag_tub         1.000000000    0.089847816
## building_value   0.089847816    1.000000000
```

The correlation between the num_bedroom and the building value improved, otherwise it is the same.

Regression Models And Analysis

The next step in the process was to run a multiple linear regression model. The lm() function from the MASS library was utilized for this step. In this model the building value is the response variable and the other features are the predictors.

The first model I ran was the linear model code with lm() on the entire dataset to see if it would work.

Then, I viewed the summary and was dissapointed with the adjusted R_squared and residual errors.

```
##
## Call:
## lm(formula = building_value ~ ., data = complete_data25percent)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -107999881     -50449      -2994      41115  192422281
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)     -6.811e+04  4.562e+02 -149.279  < 2e-16 ***
## num_bathroom     6.888e+04  2.097e+02  328.423  < 2e-16 ***
## area_total_calc  1.135e+02  9.205e-02 1233.373  < 2e-16 ***
## num_bedroom     -3.999e+04  1.753e+02 -228.130  < 2e-16 ***
## flag_fireplace  -2.906e+04  3.718e+03   -7.817 5.42e-15 ***
## flag_tub         5.653e+04  1.038e+03   54.483  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 266800 on 2896639 degrees of freedom
## Multiple R-squared:  0.4194, Adjusted R-squared:  0.4194
## F-statistic: 4.185e+05 on 5 and 2896639 DF,  p-value: < 2.2e-16
```

I checked the AIC score for comparison with later models as described by Prabhakaran, S. (2017).

```
AIC(model_1)
```

```
## [1] 80604224
```

The next step was to create diagnostic plot of the regression model. Interpreting the plots is difficult with such a large number of observations but seems to show that the model had a fairly good fit with the data and there were no outliers outside of Cook's distance.

```
# not enough memory on computer to run during knit process
# par(mfrow=c(2,2))
# plot(model_1)
```

Create Training and Test Set

I used the code from Prabhakaran (2017) to create the training and test datasets. The training and test sets consisted of an 80 : 20 split.

I then checked the structure of the training data to confirm the correct number of rows.

```
## 'data.frame':    2317316 obs. of  6 variables:
##  $ num_bathroom   : num  3 2 1 1 3 2 1 3 1 3 ...
##  $ area_total_calc: num  3185 1401 808 750 1933 ...
```

```
##  $ num_bedroom    : num  4 3 2 2 3 4 3 5 3 4 ...
##  $ flag_fireplace : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ flag_tub       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ building_value : num  250723 81615 83097 130288 236724 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:33590] 5 27 31 34 35 36 47 54 65 67 ...
##   .. ..- attr(*, "names")= chr [1:33590] "5" "27" "31" "34" ...
```

The next step was to build the model on the training data set and view the summary.

```
##
## Call:
## lm(formula = building_value ~ ., data = trainingData)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -87581589    -52850     -5919     39403  92165738
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)     -6.255e+04  4.553e+02 -137.380   <2e-16 ***
## num_bathroom     8.074e+04  2.091e+02  386.219   <2e-16 ***
## area_total_calc  9.204e+01  9.001e-02 1022.485   <2e-16 ***
## num_bedroom     -3.791e+04  1.750e+02 -216.690   <2e-16 ***
## flag_fireplace  -3.377e+04  3.727e+03   -9.062   <2e-16 ***
## flag_tub         6.805e+04  1.037e+03   65.649   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 238200 on 2317310 degrees of freedom
## Multiple R-squared:  0.4116, Adjusted R-squared:  0.4116
## F-statistic: 3.242e+05 on 5 and 2317310 DF,  p-value: < 2.2e-16
```

The results from the this model are shown above. The Adjusted R-square is 0.4359. This model explains 43.6% of the variation in the dependent variable (Lantz, 2015). I had hoped this value to be higher. The majority of the predictions are between the 1st and 3rd quartile, and were between 36,507 over the true value and 32,351 under the true value (Lantz, 2015). The overall model appears to be statistically significant (Prabhakaran 2017) as evidenced by the p values for each predictor and the F-statistic for the model which are less than the significance level of 0.05.

I then ran the AIC and BIC scores to compare with the first model.

```
AIC(model_2)
```

```
## [1] 63956461
```

The AIC score improved.

```
BIC(model_2)
```

```
## [1] 63956550
```

Next, I ran the model on test data to get the predictions.

```
building_valuePred <- predict(model_2, testData)
```

The next step was to look at the accuracy of the predicted values. According to Prabhakaran (2017), there are several measures which can be used to look at prediction accuracy. Two of them are shown here. The first measure is the correlation between the actual values and the predicted values. According to this source, a good correlation "implies that the actual values and the predicted values show similar directional movement"

(Prabhakaran, 2017) , which is seen here. The first step was to make a dataframe with the actual test values of building_value and the predicted building_values.

```
actuals_preds <- data.frame(cbind(actuals=testData$building_value, predicteds=building_valuePred))
```

```
correlation_accuracy <- cor(actuals_preds)
head(actuals_preds)
```

```
##    actuals predicteds
## 3   193796   100902.7
## 9    32654   191465.9
## 12  197599   310468.2
## 21  309226   229200.6
## 30   51702   335041.7
## 37  130000   160818.0
```

The correlation accuracy as shown below.

```
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.7122072
## predicteds 0.7122072  1.0000000
```

The above correlation between the actual values and the predicted is considered good by Lantz (2015).

The second value is the Min Max Accuracy. The closer the higher the score the better according to Prabhakaran (2017). The Min Max Accuracy score was 0.71.

Next, I checked for multi-colinearity of the regression model using the vif function:

```
##    num_bathroom area_total_calc     num_bedroom  flag_fireplace
##        1.899158        1.140912        1.774810        1.000594
##        flag_tub
##        1.018374
```

There was no multi-colinearity.

Various Linear Regression Models

LM with Cross Validation

The next phase consisted of exploring different linear model functions to see if it changed the overall performance. I started with the lm with cross Validation as suggested by Prabhakaran (2017). I chose to use the Caret package for the lm with cross validation.

The following code for the linear model with cross validation was taken from Datacamp (2016) at https://www.youtube.com/watch?v=OwPQHmiJURI.

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -87581589    -52850     -5919     39403  92165738
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
```

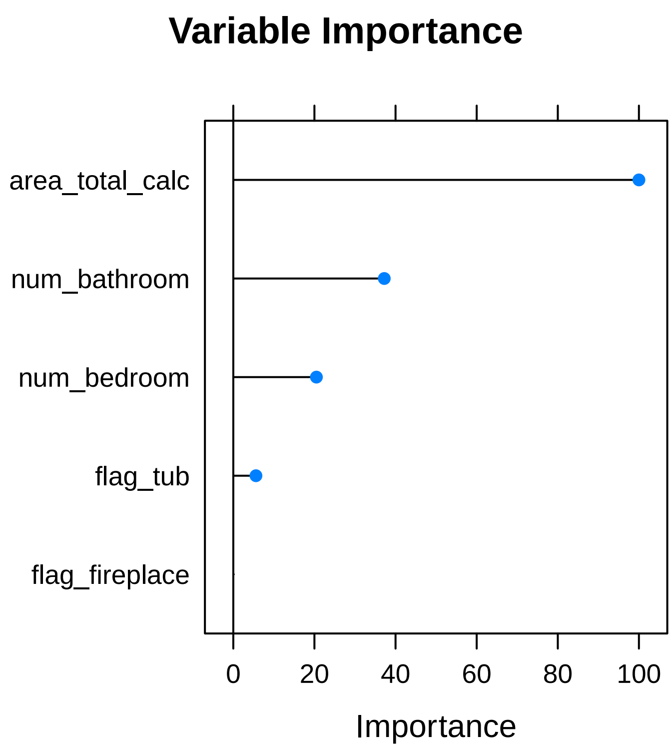
```
## (Intercept)     -6.255e+04  4.553e+02 -137.380   <2e-16 ***
## num_bathroom     8.074e+04  2.091e+02  386.219   <2e-16 ***
## area_total_calc  9.204e+01  9.001e-02 1022.485   <2e-16 ***
## num_bedroom     -3.791e+04  1.750e+02 -216.690   <2e-16 ***
## flag_fireplace  -3.377e+04  3.727e+03   -9.062   <2e-16 ***
## flag_tub         6.805e+04  1.037e+03   65.649   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 238200 on 2317310 degrees of freedom
## Multiple R-squared:  0.4116, Adjusted R-squared:  0.4116
## F-statistic: 3.242e+05 on 5 and 2317310 DF,  p-value: < 2.2e-16
```

The results of the above model were consistant with the previous models. Next, I looked at the variable importance for the above model since it was within the caret package.

Variable Importance

One of my objectives for this project was to identify which features were most important in the prediction model. This was achieve with the variable importance function in the caret package as described by Kaushik (2016). The results showed that the Total finished Living Area was the most important feature followed by the number of bathrooms and then the number of bedrooms. Hot tubs and fireplaces were, as expected for the warm climate of Los Angeles, very low value predictors.

```
## lm variable importance
##
##                 Overall
## area_total_calc 100.000
## num_bathroom     37.216
## num_bedroom      20.488
## flag_tub          5.584
## flag_fireplace    0.000
```



GLMNET

Next, I decided to try the glmnet() in Caret to see if this model performed better. This is a generalized linear model with regularization techniques.

```
## glmnet
##
## 2317316 samples
##       5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2085585, 2085584, 2085584, 2085584, 2085584, 2085585, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      RMSE      Rsquared   MAE
##   0.10     378.0916  231176.2  0.4438438  71006.74
##   0.10    3780.9159  231178.1  0.4438521  70741.49
##   0.10   37809.1587  232614.6  0.4406844  69009.07
##   0.55     378.0916  231176.8  0.4438989  70994.62
##   0.55    3780.9159  231255.7  0.4439597  70220.98
##   0.55   37809.1587  235961.8  0.4303314  70230.47
##   1.00     378.0916  231179.3  0.4439143  70985.87
##   1.00    3780.9159  231392.7  0.4437921  69782.80
##   1.00   37809.1587  238370.0  0.4300008  70773.63
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0.1 and lambda = 378.0916.
```

The above summary showed similar RMSE and R-aquared values.

Improve Model

Removing Features

I decided to experiment with the dataset to see if I could improve the performance by altering the number of features in the dataset. I first tried removing features one by one to see if it improves the model. I decided to use the Caret package for this. The following code was taken from Datacamp (2016) at https://www.youtube.com/watch?v=OwPQHmiJURI.

```
model4_caret <- train(building_value ~ area_total_calc + num_bedroom + num_bathroom, trainingData, meth
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -87827161    -52865     -6121     39400  92085669
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6.384e+04  4.552e+02  -140.2   <2e-16 ***
## area_total_calc  9.229e+01  9.002e-02  1025.3   <2e-16 ***
## num_bedroom     -3.804e+04  1.751e+02  -217.3   <2e-16 ***
## num_bathroom     8.196e+04  2.084e+02   393.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 238400 on 2317312 degrees of freedom
## Multiple R-squared:  0.4105, Adjusted R-squared:  0.4105
## F-statistic: 5.378e+05 on 3 and 2317312 DF,  p-value: < 2.2e-16
```

The above model's performance was worse.

I also used the glmnet package in caret on with the 3 independent variables

```
glmnet2 <- train(building_value ~ area_total_calc + num_bedroom + num_bathroom, trainingData, method =
```

```
## glmnet
##
## 2317316 samples
##       3 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2085584, 2085586, 2085585, 2085584, 2085585, 2085584, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      RMSE      Rsquared   MAE
##   0.10     378.0916  239051.7  0.4432465  71239.42
##   0.10    3780.9159  239034.6  0.4432808  70974.13
##   0.10   37809.1587  239869.6  0.4404317  69273.97
##   0.55     378.0916  239083.4  0.4432931  71219.50
```

```
##   0.55    3780.9159  239115.6  0.4434365  70469.96
##   0.55   37809.1587  242925.7  0.4324981  70298.41
##   1.00     378.0916  239089.8  0.4433039  71221.99
##   1.00    3780.9159  239248.7  0.4433716  70048.53
##   1.00   37809.1587  245374.8  0.4331248  70846.33
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0.1 and lambda = 3780.916.
```

Next, I ran model with 2 independent variables, just to experiment.

```
model5_caret <- train(building_value ~ area_total_calc + num_bedroom, trainingData, method = "lm", trCon
```

```
## Linear Regression
##
## 2317316 samples
##       2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2085584, 2085584, 2085585, 2085585, 2085585, 2085584, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   246137.1  0.4022237  72935.51
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Less features results in higher RMSE and Rsquared, which is not improving the model. It was time to try adding more features instead.

Adding Features

I redefined my sample set to include a few more features. I went back to see which features contained NAs between 25% and 50% and found three more I could add into a new sample set. Those three were the type of home heating, the quality or condition of the building and the number of units built into the structure such as a duplex or triplex. I added heating, num-unit, quality into a new dataframe to try to improve the model. Then, I repeated the process for cleaning the second data set.

```
dataset_2 <- struct_features_only_data[ , c('num_bathroom', 'area_total_calc', 'num_bedroom', 'flag_fire
```

```
table(dataset_2$flag_fireplace)
```

```
##
##            true
## 2925072    5163
```

Checking correlation of new data set

```
##                  num_bathroom area_total_calc num_bedroom flag_fireplace
## num_bathroom      1.000000000      0.35339107  0.68012096  -0.0050893415
## area_total_calc   0.353391072      1.00000000  0.26186610  -0.0090968699
## num_bedroom       0.680120957      0.26186610  1.00000000  -0.0189496968
## flag_fireplace   -0.005089341     -0.00909687 -0.01894970   1.0000000000
## flag_tub          0.127130815      0.08404231  0.07647492  -0.0031970910
## building_value    0.273515348      0.62947348  0.12792031  -0.0064059790
## heating          -0.369716753     -0.28195472 -0.22521453   0.0377887188
```

```
## num_unit         0.037146721      0.38655740  0.05487475   0.0004126424
## quality         -0.267979423     -0.13957964 -0.04405096   0.0006847953
##                    flag_tub building_value      heating      num_unit
## num_bathroom     0.127130815    0.273515348 -0.369716753  0.0371467211
## area_total_calc  0.084042306    0.629473481 -0.281954724  0.3865573994
## num_bedroom      0.076474921    0.127920306 -0.225214534  0.0548747520
## flag_fireplace  -0.003197091   -0.006405979  0.037788719  0.0004126424
## flag_tub         1.000000000    0.077746744 -0.032236250 -0.0071203615
## building_value   0.077746744    1.000000000 -0.188601923  0.2618926080
## heating         -0.032236250   -0.188601923  1.000000000  0.0059192784
## num_unit        -0.007120361    0.261892608  0.005919278  1.0000000000
## quality         -0.047842484   -0.114141649  0.434391349  0.0959453518
##                      quality
## num_bathroom    -0.2679794232
## area_total_calc -0.1395796367
## num_bedroom     -0.0440509610
## flag_fireplace   0.0006847953
## flag_tub        -0.0478424839
## building_value  -0.1141416492
## heating          0.4343913492
## num_unit         0.0959453518
## quality          1.0000000000
```

A summary of the data set at this point is shown below.

```
##   num_bathroom    area_total_calc   num_bedroom     flag_fireplace
##  Min.   : 0.000   Min.   :     1   Min.   : 0.000   Min.   :0.000000
##  1st Qu.: 2.000   1st Qu.:  1217   1st Qu.: 2.000   1st Qu.:0.000000
##  Median : 2.000   Median :  1579   Median : 3.000   Median :0.000000
##  Mean   : 2.239   Mean   :  1834   Mean   : 3.131   Mean   :0.001762
##  3rd Qu.: 3.000   3rd Qu.:  2145   3rd Qu.: 4.000   3rd Qu.:0.000000
##  Max.   :20.000   Max.   :952576   Max.   :20.000   Max.   :1.000000
##  NA's   :23       NA's   :33574    NA's   :12
##     flag_tub        building_value         heating            num_unit
##  Min.   :0.00000   Min.   :        1   Min.   : 1         Min.   :  1.0
##  1st Qu.:0.00000   1st Qu.:    74800   1st Qu.: 2         1st Qu.:  1.0
##  Median :0.00000   Median :   122590   Median : 2         Median :  1.0
##  Mean   :0.02355   Mean   :   170884   Mean   : 4         Mean   :  1.2
##  3rd Qu.:0.00000   3rd Qu.:   196889   3rd Qu.: 7         3rd Qu.:  1.0
##  Max.   :1.00000   Max.   :251486000   Max.   :24         Max.   :997.0
##                                        NA's   :1126033   NA's   :956108
##     quality
##  Min.   : 1.0
##  1st Qu.: 4.0
##  Median : 7.0
##  Mean   : 5.8
##  3rd Qu.: 7.0
##  Max.   :12.0
##  NA's   :994925
```

Next, I identified number of complete cases.

```
sum(complete.cases(dataset_2))
```

```
## [1] 1734522
```

Then, I removed incomplete cases from the second data set.

```r
dataset_2_complete <- na.omit(dataset_2)
```

I checked the summary of the second data set.

```
##   num_bathroom  area_total_calc  num_bedroom    flag_fireplace
##  Min.   : 0.0   Min.   :    1   Min.   : 0.00   Min.   :0.0e+00
##  1st Qu.: 2.0   1st Qu.: 1166   1st Qu.: 2.00   1st Qu.:0.0e+00
##  Median : 2.0   Median : 1494   Median : 3.00   Median :0.0e+00
##  Mean   : 2.2   Mean   : 1706   Mean   : 3.06   Mean   :1.2e-06
##  3rd Qu.: 3.0   3rd Qu.: 1983   3rd Qu.: 4.00   3rd Qu.:0.0e+00
##  Max.   :20.0   Max.   :44657   Max.   :20.00   Max.   :1.0e+00
##     flag_tub      building_value       heating          num_unit
##  Min.   :0.0000   Min.   :       9   Min.   : 2.000   Min.   :  1.000
##  1st Qu.:0.0000   1st Qu.:   75609   1st Qu.: 2.000   1st Qu.:  1.000
##  Median :0.0000   Median :  120504   Median : 2.000   Median :  1.000
##  Mean   :0.0114   Mean   :  163417   Mean   : 3.734   Mean   :  1.003
##  3rd Qu.:0.0000   3rd Qu.:  189372   3rd Qu.: 7.000   3rd Qu.:  1.000
##  Max.   :1.0000   Max.   :26760000   Max.   :20.000   Max.   :143.000
##     quality
##  Min.   : 1.000
##  1st Qu.: 4.000
##  Median : 7.000
##  Mean   : 5.672
##  3rd Qu.: 7.000
##  Max.   :12.000
```

A training set and a test set were created with an 80:20 split.

Linear Regression on trainingData2

Once the second data set was cleaned I was ready to run the regression models again using the same functions as I did with initial data set.

```
##
## Call:
## lm(formula = building_value ~ ., data = trainingData2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5099162   -48904    -1775    40595 25572251
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)     -1.105e+05  7.921e+02 -139.557  < 2e-16 ***
## num_bathroom     4.212e+04  1.960e+02  214.874  < 2e-16 ***
## area_total_calc  1.552e+02  2.113e-01  734.467  < 2e-16 ***
## num_bedroom     -4.173e+04  1.451e+02 -287.647  < 2e-16 ***
## flag_fireplace  -4.362e+03  8.864e+04   -0.049    0.961
## flag_tub         5.243e+03  1.005e+03    5.216 1.83e-07 ***
## heating         -8.395e+02  5.197e+01  -16.152  < 2e-16 ***
## num_unit        -5.958e+03  5.506e+02  -10.821  < 2e-16 ***
## quality          9.398e+03  6.527e+01  143.990  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 125400 on 1387608 degrees of freedom
## Multiple R-squared:  0.5696, Adjusted R-squared:  0.5696
## F-statistic: 2.296e+05 on 8 and 1387608 DF,  p-value: < 2.2e-16
```

I noted some improvement in the RSE and Adjusted R-square. The features, flag_fireplace and num_unit are not significant predictors, so I removed them from the data set and updated the train and test set.

The structure of new training set.

```
## 'data.frame':    1387617 obs. of  7 variables:
##  $ num_bathroom   : num  2 2 2 1 3 3 3 2 3 2 ...
##  $ area_total_calc: num  1240 1373 1257 966 2423 ...
##  $ num_bedroom    : num  3 4 2 2 4 3 3 3 5 2 ...
##  $ flag_tub       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ building_value : num  35681 28335 111942 73172 163377 ...
##  $ heating        : int  7 7 7 7 2 2 2 7 2 2 ...
##  $ quality        : int  7 7 7 7 7 4 7 7 4 7 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:1195713] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..- attr(*, "names")= chr [1:1195713] "1" "2" "3" "4" ...
```

Next, I ran the lm() on the revised dataset.

```
##
## Call:
## lm(formula = building_value ~ ., data = trainingData3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5096092   -48900    -1776    40592 25572563
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)     -1.164e+05  5.752e+02 -202.409  < 2e-16 ***
## num_bathroom     4.214e+04  1.960e+02  214.983  < 2e-16 ***
## area_total_calc  1.551e+02  2.111e-01  734.560  < 2e-16 ***
## num_bedroom     -4.171e+04  1.451e+02 -287.529  < 2e-16 ***
## flag_tub         5.270e+03  1.005e+03    5.242 1.59e-07 ***
## heating         -8.467e+02  5.197e+01  -16.293  < 2e-16 ***
## quality          9.397e+03  6.527e+01  143.965  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 125400 on 1387610 degrees of freedom
## Multiple R-squared:  0.5696, Adjusted R-squared:  0.5696
## F-statistic: 3.061e+05 on 6 and 1387610 DF,  p-value: < 2.2e-16
```

The results of the lm model haven't changed with the removal of the flag_fireplace and num_unit.

Calcualte Metrics

Calcuating AIC and BIC on the model with 6 predictors.

```r
AIC(fit_C)
```

```
## [1] 36516218
```

```r
BIC(fit_C)
```

```
## [1] 36516316
```

AIC and BIC values improved compared to initial dataset with 5 predictors.

The next step was to run the model on testData3 set to get predictions.

```r
building_valuePred3 <- predict(fit_C, testData3)
```

```r
summary(building_valuePred3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -242588   79271  130866  163358  204256 5249864
```

The prediction accuracy metrics were then calcuated for a comparison.

The correlation accuracy improved to 0.70 and the Min Max Accuracy improved to 0.72,

Check the min_max_accuracy.

LM with Cross Validation on 6 Predictors

The following step is to check model performance using lm with cross validation on second dataset.

```r
Fit_B_lm <- train(building_value ~ ., trainingData3, method = "lm", trControl = trainControl(method = "
```

```
## Linear Regression
##
## 1387617 samples
##       6 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1248856, 1248856, 1248854, 1248855, 1248855, 1248856, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   125137.8  0.5711848  65627.61
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

The above results are Similar results to lm().

I also looked for muti-colinearity

```
##    num_bathroom area_total_calc     num_bedroom        flag_tub
##        3.238311        3.038271        1.769576        1.004121
##         heating         quality
##        1.452097        1.277864
```

There is no multi-colinearity.

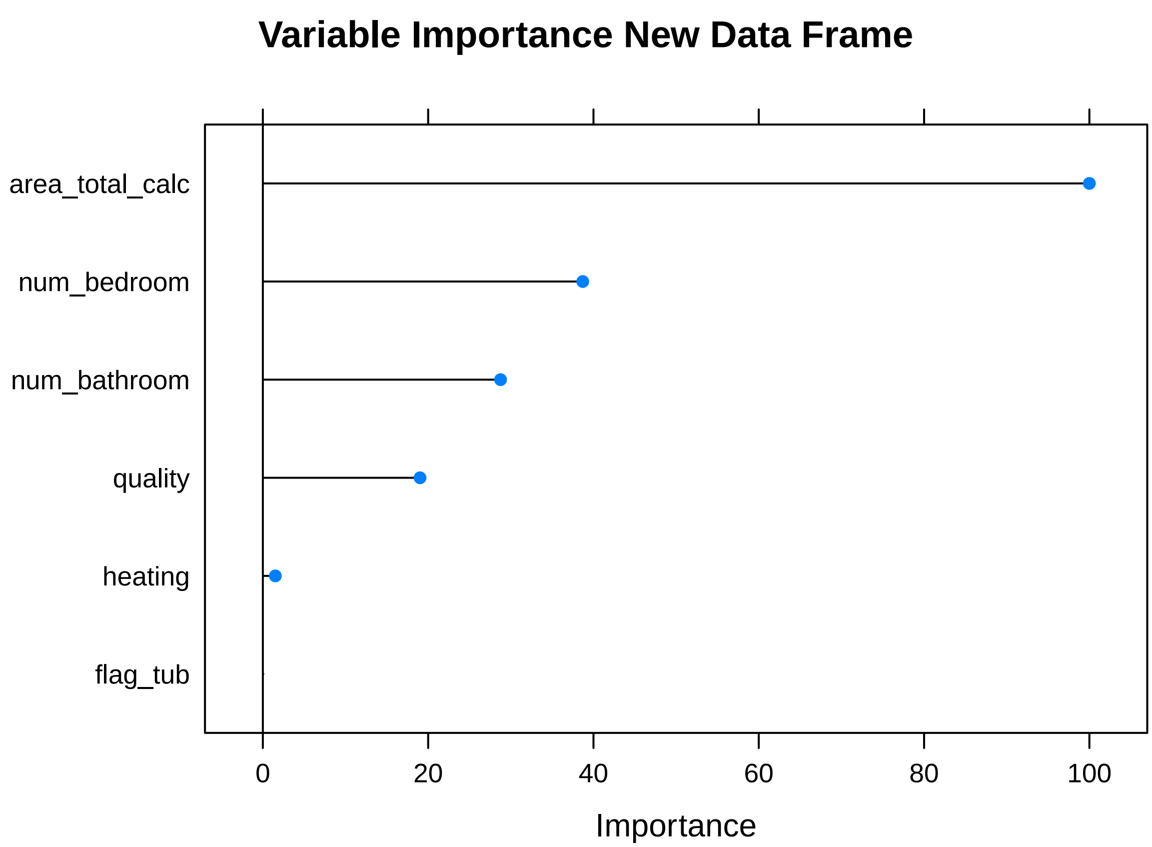
I also ran the glmnet function on second dataset and the results were similar.

Results are consistent.

I rechecked the variable importance on the model from the second data set and created a plot.

Statistic	Criterion	5 predictors	6 predictors
Adj R – Square	Higher the better	0.44	0.49
F – Statistic	Higher the better	3.232e+05	2.033e+05
Residual Standard Error	Lower the better	58,780	54,700
AIC	Lower the better	51,879,664	31,679,908
BIC	Lower the better	51,879,752	31,680,004
MinMaxAccuracy	Higher the better	0.71	0.72
Correlation	Higher the better	0.66	0.70

Figure 1: Comparison of Regression Models.



The above plot is similar to the first variable importance plot where area_total_calc has the highest importance, followed by the quality feature and then the num_bathroom.

Discussion

In conclusion, a multiple linear regression model was created to predict the value of a home. The results of the model improved when more features were added to the dataset. The type of linear function utlized did not alter the performance of the model, but note that the tuning parameters were not adjusted. The model with the best performance was the one with the second data set which had 6 predictors. However, this model requires more work before it can be published for proprietary use as the performance is not satifactory.

Several options are available which may help improve this prediction model furhter. First, You could continuing to add more features, but by including more structural features there will be more NAs to deal with. Also, consideration should be give to imputing values or removing more rows with NAs since there is a large number of observations. Next, manipulating the tuning parameters within regression model fucntion maybe improve results. Another option is to find a better data set with fewer missing values. Lastly, to make this model more useful throughout similar geographical areas, additional real estate data from other geographical areas could be obtained and aggregated with the Los Angeles Data.

References

Clemens (2018) Retrieved from https://stackoverflow.com/questions/48658832/how-to-remove-row-if-it-has-a-na-value-one-cer

Datacamp (2016) Retrieved from https://www.youtube.com/watch?v=OwPQHmiJURI

Dhana, K. (2016). Identify, describe, plot, and remove the outliers from the dataset. Retrieved from https://www.r-bloggers.com/identify-describe-plot-and-remove-the-outliers-from-the-dataset/y

James (2011) Retrieved from https://stackoverflow.com/questions/7567790/change-the-index-number-of-a-dataframe;

Kaushik, S. (2016). Practical guide to implement machine learning with CARET package in R (with practice problem). Analytics Vidhya. Retrieved from https://www.analyticsvidhya.com/blog/2016/12/practical-guide-to-implement-machine-learning-with-caret-package-in-r-with-practice-problem/

Lantz, B. (2015). Chapter 6: Forecasting Numeric Data - Regression Methods. In B. Lantz, Machine Learning with R Second Edition - Classification Using Decision Trees and Rules (pp. 171-218). Packt Publishing.

Prabhakaran, S. (2017). Linear Regression. R-statistics.co. Retrieved from http://r-statistics.co/Linear-Regression.html

RDocumentation (n.d.) Retrieved from https://www.rdocumentation.org/packages/stats/versions/3.5.0/topics/cor.

Spachtholz (2017) Retrieved from https://www.kaggle.com/philippsp/exploratory-analysis-zillow

Subsetting R data frame results in mysterious NA rows (n.d) Retrieved from https://stackoverflow.com/questions/14261619/subsetting-r-data-frame-results-in-mysterious-na-rows

Walters, T. (2017). A very extensive Zillow exploratory analysis. Retrieved from https://www.kaggle.com/captcalculator/a-very-extensive-zillow-exploratory-analysis

Zillow Prize: Zillow's Home Value Prediction(Zestimate) (2018). Properties_2016. Retrieved from https://www.kaggle.com/c/zillow-prize-1/data

Appendix

I ran the lm() with log of response variable as shown in the steps below. However, I am not as compfortable manipulating or interpreting the log value, so I did not include it in the discussion.

```
m2 <- lm(log(building_value) ~ ., data = trainingData)
```

```r
summary(m2)
```

```
##
## Call:
## lm(formula = log(building_value) ~ ., data = trainingData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.503  -0.314   0.061   0.388   6.024
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.069e+01  1.203e-03 8883.33   <2e-16 ***
## num_bathroom     4.746e-01  5.525e-04  859.00   <2e-16 ***
## area_total_calc  7.703e-05  2.379e-07  323.81   <2e-16 ***
## num_bedroom     -6.659e-02  4.624e-04 -144.01   <2e-16 ***
## flag_fireplace  -1.153e-01  9.851e-03  -11.71   <2e-16 ***
## flag_tub         3.556e-01  2.739e-03  129.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6295 on 2317310 degrees of freedom
## Multiple R-squared:  0.4037, Adjusted R-squared:  0.4037
## F-statistic: 3.138e+05 on 5 and 2317310 DF,  p-value: < 2.2e-16
```

Residuals may have improved the median is closer to zero, Residual Standar error is lower, but also the Adjusted R-squared. Not sure how to interpret this.

Run log model on test set

```r
building_valuePred_log <- predict(m2, testData)
```

Calculate prediction accuracy Make a dataframe with the actual test values of building_value and the predicted building_values

```r
actuals_preds_log <- data.frame(cbind(actuals=testData$building_value, predicteds=building_valuePred_log
```

```r
correlation_accuracy_log <- cor(actuals_preds_log)
head(actuals_preds_log)
```

```
##    actuals predicteds
## 3   193796   10.82688
## 9    32654   10.90268
## 12  197599   11.00228
## 21  309226   10.93427
## 30   51702   11.02285
## 37  130000   10.87703
```

```r
correlation_accuracy_log
```

```
##              actuals predicteds
## actuals    1.0000000  0.4055539
## predicteds 0.4055539  1.0000000
```

Using the log did not have any better than previous models.