Econ 104 Project 1

Matthew Craig, Zain Gaffer, Kento Koguchi

2023-01-26

Contents

```
rm(list=ls(all=TRUE))
library(Boruta)
library(car)
library(caret)
library(corrplot)
library(dplyr)
library(ggplot2)
library(leaps)
library(lubridate)
library(lmtest)
library(lmvar)
library(randomForest)
library(RColorBrewer)
library(sandwich)
library(stargazer)
library(tidyr)
```

Dataset

For this project, we chose a dataset containing House Sales in King County, WA between May 2014 and May 2015. The dataset is available from kaggle here. The dataset contains 21 columns, including our response variable `price`.

```
kc_house <- read.csv("kc_house_data.csv", header=TRUE)
```

Column Descriptions:

- `id`: Unique ID for each home sold
- `date`: Date the home was sold
- `price`: Price of the home sold
- `bedrooms`: Number of bedrooms
- `bathrooms`: Number of bathrooms (including 0.5 and 0.75 values for half and three-quarters bathrooms)

- `sqft_living`: Square footage of the interior living space
- `sqft_lot`: Square footage of the property
- `floors`: Number of floors in the home
- `waterfront`: An indicator variable for whether the home is located next to the waterfront
- `view`: An index from 0 to 4 corresponding to a subjective assessment of the property view
  - Coding: 0 = No view, 1 = Fair, 2 = Average, 3 = Good, 4 = Excellent
- `condition`: An index from 1 to 5 corresponding to a subjective assessment of the property condition
  - Coding: 1 = Poor-Worn out, 2 = Fair-Badly worn, 3 = Average, 4 = Good, 5 = Very Good
- `grade`: An index from 1 to 13, where 1-3 falls short of acceptable building construction and design, 7 has an average level of construction and design, and 11-13 has a high quality level of construction and design.
- `sqft_above`: Square footage of the interior living space that is above ground level
- `sqft_basement`: Square footage of the interior living space that is below ground level
- `yr_built`: The year the house was initially built
- `yr_renovated`: The year of the house's last renovation
- `zipcode`: What zipcode the property is located in
- `lat`: Latitude
- `long`: Longitude
- `sqft_living15`: The average square footage of the interior living space for the nearest 15 neighbors
- `sqft_lot15`: The average square footage of the property lots of the nearest 15 neighbors

Data Transformations

For this project, we apply a few transformations to the raw dataset to make it appropriate for use in a linear regression model:

```
# Drop ID column, since it's just an identifier
kc_house <- subset(kc_house, select=-c(id))

# Drop zipcode, since there are too many categories
kc_house <- subset(kc_house, select=-c(zipcode))

# Extract Date object from date column
kc_house$date_sold <- as.Date(kc_house$date, "%Y%m%d")
# Extract year sold (as an indicator variable)
kc_house$yr_sold <- year(kc_house$date_sold) - 2014
# Extract month sold
kc_house$mnth_sold <- month(kc_house$date_sold)
# Convert to "season" indicator
kc_house$spr_sum <- 0
kc_house[kc_house$mnth_sold >= 3 & kc_house$mnth_sold <= 8, 'spr_sum'] <- 1
# Drop date sold and month
kc_house <- subset(kc_house, select=-c(date, date_sold, mnth_sold))

# Fill in zero values in yr_renovated
kc_house[kc_house$yr_renovated == 0, 'yr_renovated'] <-
  kc_house[kc_house$yr_renovated == 0, 'yr_built']

# Normalize columns
kc_house$condition <- kc_house$condition - 3
kc_house$grade <- kc_house$grade - 7
```

```
kc_house$yr_built <- kc_house$yr_built - 1900
kc_house$yr_renovated <- kc_house$yr_renovated - 1900
kc_house$long <- abs(kc_house$long) - 121
kc_house$lat <- kc_house$lat - 47
```

A summary of the data transformations:

- Drop the `id` column, since it is an identifier and not a predictor
- Drop the `zipcode` column, since it is actually categorical and not numeric. While the old real estate adage is that the most important factors are "Location, Location, Location!", as economists, it would be difficult for us to assign an economic interpretation to a numerical coefficient for zipcode.
- Extract the year the home was sold from the `date` string. Since there are only two years in the dataset (2014 and 2015), convert the year into an indicator variable `yr_sold` with 0 = 2014 and 1 = 2015.
- Extract the month the home was sold from the `date` string. Months are not ordinal but categorical; however, since we already have many predictor variables, we don't want to add 11 more to represent each month. Instead, we convert the month into a single indicator variable `spr_sum` that denotes whether the home was sold in the spring/summer (March-August) or not. The choice of which months to group was somewhat arbitrary.
- Impute the zero values of `yr_renovated`. Since some houses were never renovated, they have a value of zero. However, this is more appropriately a NA value. In effect, this column indicates the last year the home was upgraded, so a reasonable imputation is the year the home was built.
- Normalize the columns `condition` and `grade` by subtracting the average value (3 and 7, respectively). This means the average house should have 0 condition and grade, which makes the economic interpretation of the coefficient simpler.
- Normalize the columns `yr_built`, `yr_renovated`, `long`, and `lat` by subtracting the minimum value. This should result in the minimum value being zero, which has a better economic interpretation, and prevents the coefficients of the regression from growing too large.

In the end, we dropped 3 columns (`id`, `date`, and `zipcode`) and created two more (`yr_sold` and `spr_sum`), leaving 19 predictor variables. All the columns in our dataset are now numerical.

Note: While we did center some columns like `condition` and `grade`, we declined to center or scale columns like `sqft_living` or `sqft_lot`, despite their very large values, because the distributions are far from normal and we wanted to maintain the economic interpretation of the coefficients at the scale of one square foot.

We also decline to transform columns or `price`, for example, with a log or Box-Cox transformation to improve prediction accuracy, since the assignment did not call for extensive pre-processing.

1. Descriptive Analysis

Statistical Summary

```
summary_df <- sapply(kc_house, fivenum)
rownames(summary_df) <- c("Min", "Q1", "Median", "Q3", "Max")
summary_df <- t(summary_df) # Transpose to fit on page
stargazer(summary_df, type="text")
```

Five Number Summary

```
##
## ===================================================
##                 Min     Q1    Median    Q3       Max
## ---------------------------------------------------
## price         75,000 321,950 450,000 645,000 7,700,000
## bedrooms        0       3       3       4       33
## bathrooms       0     1.750   2.250   2.500      8
## sqft_living    290    1,427   1,910   2,550   13,540
## sqft_lot       520    5,040   7,618  10,688  1,651,359
## floors          1       1     1.500     2      3.500
## waterfront      0       0       0       0        1
## view            0       0       0       0        4
## condition       -2      0       0       1        2
## grade           -6      0       0       1        6
## sqft_above     290    1,190   1,560   2,210    9,410
## sqft_basement   0       0       0      560     4,820
## yr_built        0      51      75      97       115
## yr_renovated    0      54      77      99       115
## lat           0.156   0.471   0.572   0.678    0.778
## long          0.315   1.125   1.230   1.328    1.519
## sqft_living15  399    1,490   1,840   2,360    6,210
## sqft_lot15     651    5,100   7,620  10,083   871,200
## yr_sold         0       0       0       1        1
## spr_sum         0       0       1       1        1
## ---------------------------------------------------
```

We can see reasonable values for most columns of the dataset. The most expensive house was over $7M, while the least was less than $100k. Most houses cost around half a million dollars, have 3-4 bedrooms and about 2 or 2.5 bathrooms on 1-2 floors (no or small basement), and range from 1500-2500 square feet in size. Most houses are of average condition and grade (centered to 0), with no view, and not located on the waterfront. The values for `sqft_living15` and `sqft_lot15` indicate that most houses are located near other houses which are also of typical size. All of these values seem very reasonable for a typical American suburban home.

The oldest house in the dataset is from 1900, with most houses having been built in the mid-late 19th century. The summary statistics provide no useful information for `lat` or `long`, except perhaps that more homes are located in the north-western part of the county (where the major population centers like Seattle are).

The minimums for `bedrooms`, `bathrooms`, `sqft_living`, and `sqft_lot` indicate that there may be some houses in the dataset with missing data points, as a house with no bedrooms or no bathrooms seems exceptionally unlikely.

Likewise, the maximum values indicate some truly enormous houses, with 13k square feet of living space and 1.6M square feet of lot space. These values are so far away from the typical range of values that they may be outliers in the data, which we will explore later.

```
stargazer(kc_house, type="text")
```

Mean and Std. Dev

```
##
```

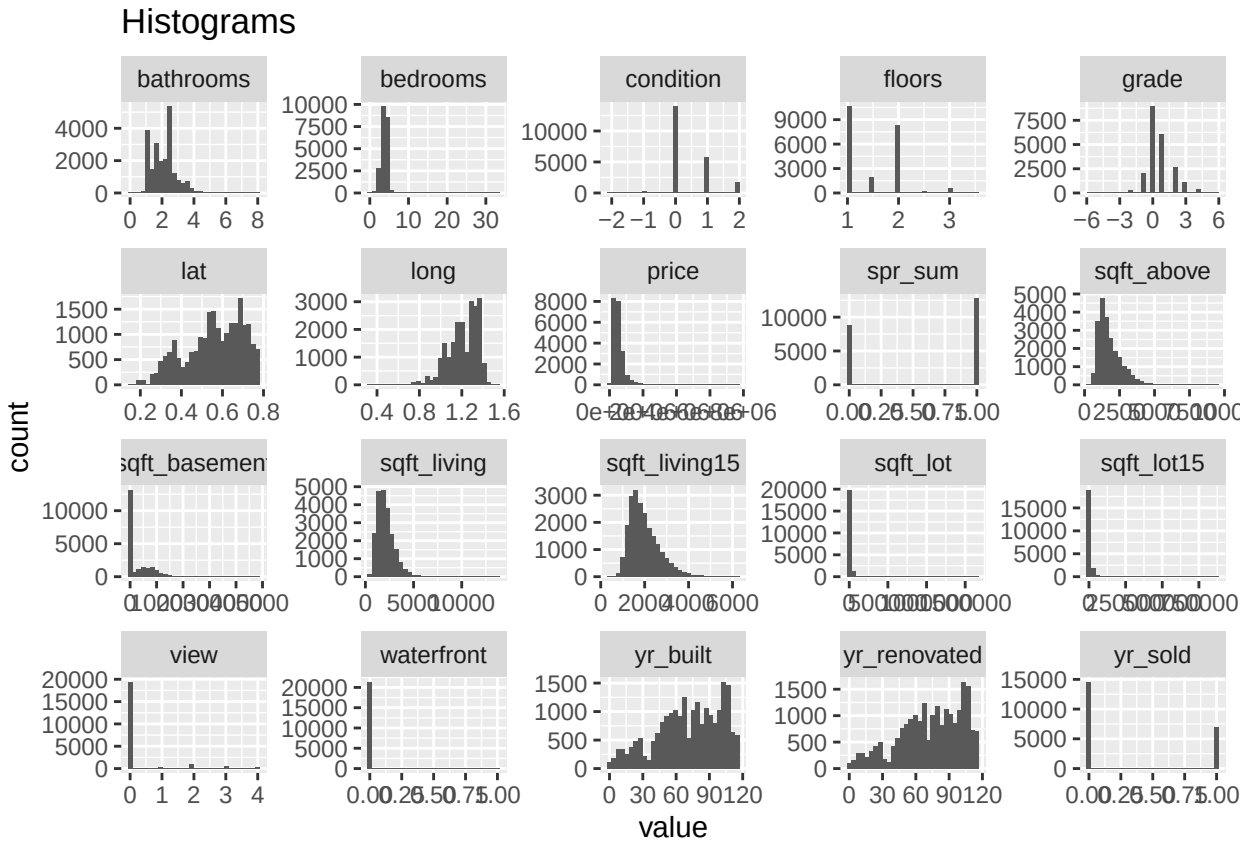
```
## ============================================================
## Statistic       N       Mean      St. Dev.    Min      Max
## ------------------------------------------------------------
## price         21,613 540,088.100 367,127.200 75,000 7,700,000
## bedrooms      21,613    3.371       0.930      0       33
## bathrooms     21,613    2.115       0.770    0.000    8.000
## sqft_living   21,613  2,079.900    918.441    290    13,540
## sqft_lot      21,613 15,106.970  41,420.510   520   1,651,359
## floors        21,613    1.494       0.540    1.000    3.500
## waterfront    21,613    0.008       0.087      0        1
## view          21,613    0.234       0.766      0        4
## condition     21,613    0.409       0.651      -2       2
## grade         21,613    0.657       1.175      -6       6
## sqft_above    21,613  1,788.391    828.091    290     9,410
## sqft_basement 21,613   291.509     442.575     0      4,820
## yr_built      21,613   71.005      29.373      0       115
## yr_renovated  21,613   73.386      28.807      0       115
## lat           21,613    0.560       0.139    0.156    0.778
## long          21,613    1.214       0.141    0.315    1.519
## sqft_living15 21,613  1,986.552    685.391    399     6,210
## sqft_lot15    21,613 12,768.460  27,304.180   651    871,200
## yr_sold       21,613    0.323       0.468      0        1
## spr_sum       21,613    0.595       0.491      0        1
## ------------------------------------------------------------
```

The mean and standard deviations of the data differ substantially from the median and IQR for some columns. The mean price is higher than the median, indicating a longer right tail on price (which is often the case for prices). `sqft_lot` has a mean nearly double the value of the median, indicating a similar effect (and a very large standard deviation), though interestingly `sqft_living` is relatively close. `condition` and `grade` are slightly right-skewed, which is expected. The mean values of the indicator variables tell us that less than 1% of houses are located on the oceanfront, only about a third of the houses were sold in 2015 (which makes sense, as the data ends in May of 2015), and slightly more than half of the houses were sold in the spring/summer.

Distributions

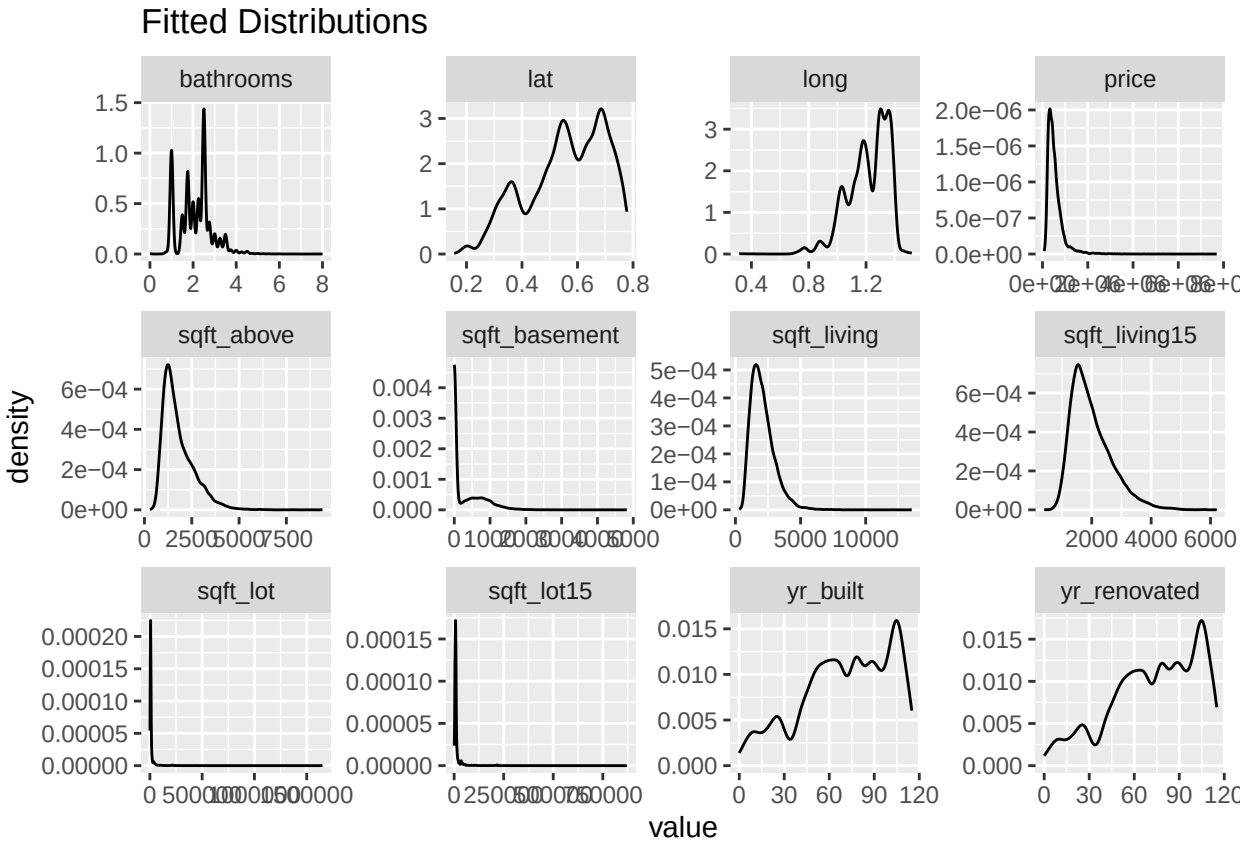
```
kc_house %>%
  tidyr::pivot_longer(everything(), names_to="key")  %>%
  ggplot(aes(x=value)) +
  geom_histogram() +
  ggtitle("Histograms") +
  facet_wrap(~ key, scales="free")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We can see from the histograms of `bedrooms`, `bathrooms`, and `floors` that moderate-sized homes are common. Most homes have an average `condition` and `grade` and no `view`. We can also see long tails on the continuous variables.

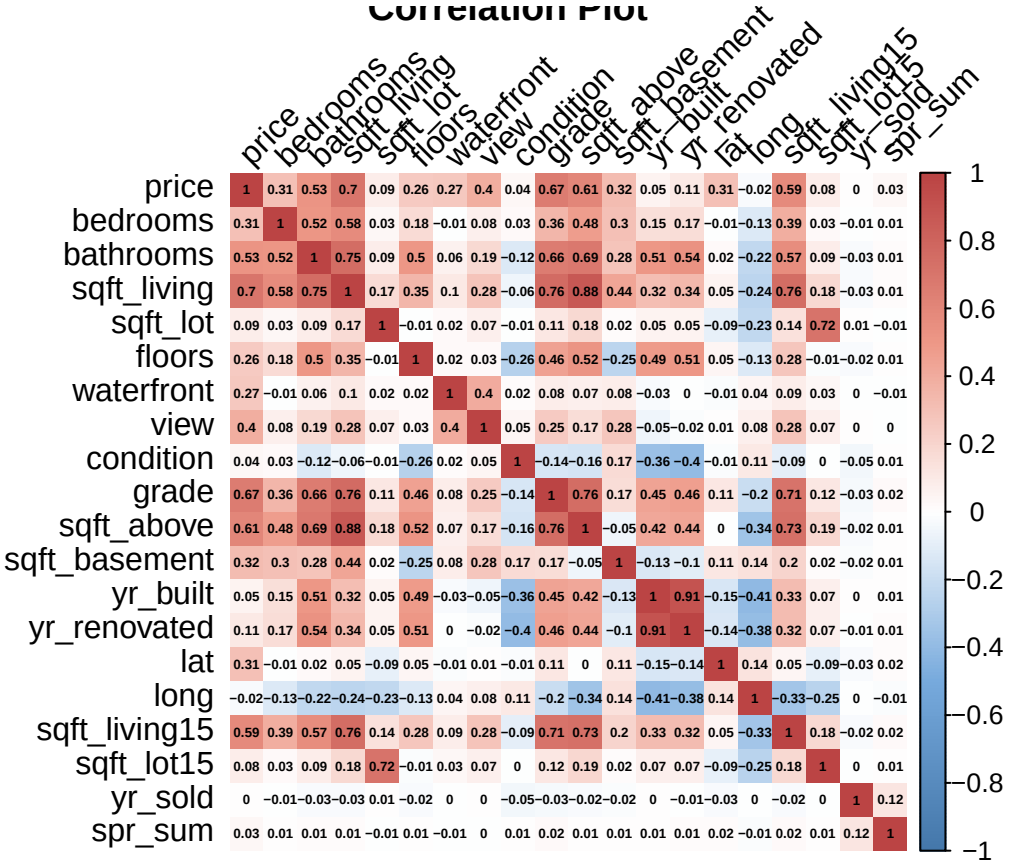
```
kc_house %>%
  tidyr::pivot_longer(-c(condition, floors, grade, spr_sum, view, waterfront, bedrooms,
                         yr_sold), names_to="key")  %>%
  ggplot(aes(x=value)) +
  geom_density() +
  ggtitle("Fitted Distributions") +
  facet_wrap(~ key, scales="free")
```



We only plot distributions for the continuous features in our dataset. `lat` and `long` indicate three large clusters of homes in our dataset. `yr_built` indicates that more houses are built recently, which is to be expected. We can see incredibly long right tails on `sqft_living`, `sqft_lot`, and `price`. These variables are far from normal. Some distribution with a long right tail would be much more appropriate, like potentially log-normal.

Correlation

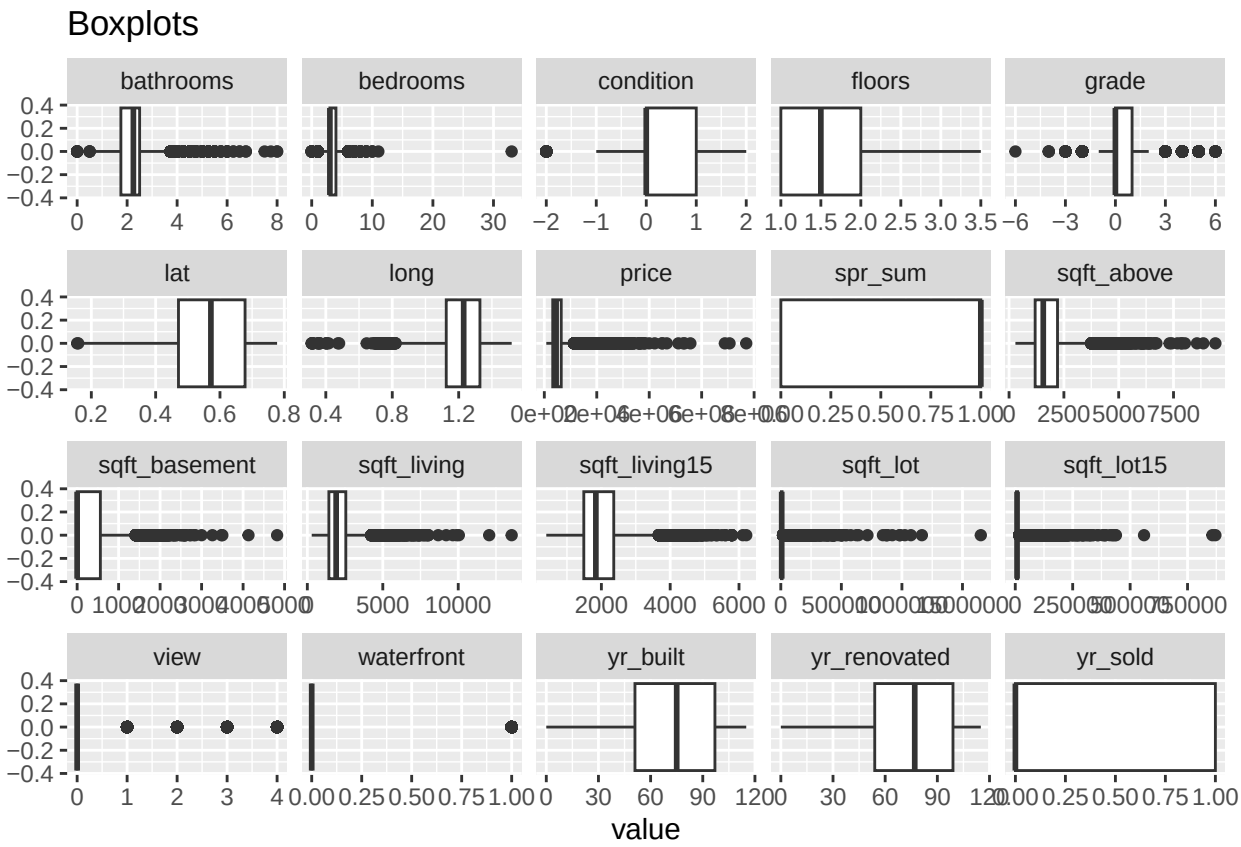
```
gradient <- colorRampPalette(c("#4477AA", "#77AADD", "#FFFFFF", "#EE9988", "#BB4444"))
corr <- cor(kc_house)
corrplot(corr, method="shade", shade.col=NA, tl.col="black", tl.srt=45,
         col=gradient(200), addCoef.col="black", title="Correlation Plot",
         number.cex=0.45)
```



We can see that quite a few columns have high correlation with `price`, like `bedrooms`, `bathrooms`, `sqft_living`, `sqft_above`, `sqft_living15`, and `grade`. These make intuitive sense as predictors of home price. We can also see that a many of those same columns are highly correlated with each other.

Boxplots

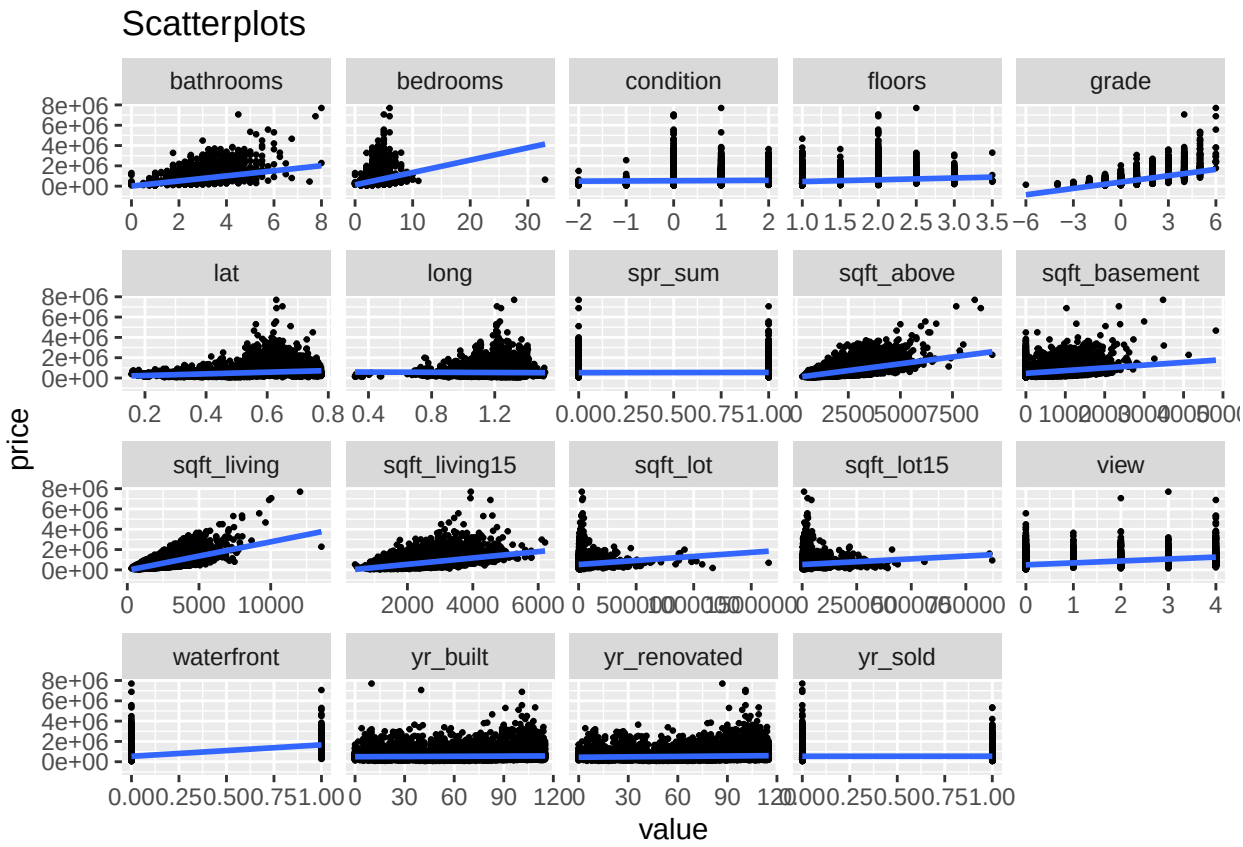
```
kc_house %>%
  tidyr::pivot_longer(everything(), names_to="key")  %>%
  ggplot(aes(x=value)) +
  geom_boxplot() +
  ggtitle("Boxplots") +
  facet_wrap(~ key, scales="free_x")
```



The boxplots confirm what our fitted distributions above indicated, that many of our features (especially `sqft_living`, `sqft_lot`, and related sqft variables) have extreme right tails. We can also see a very long right tail on `price`, which may prove to be difficult for our linear model to handle.

Scatterplots

```
kc_house %>%
  tidyr::pivot_longer(c(-price), names_to="key")  %>%
  ggplot(aes(x=value, y=price)) +
  geom_point(size=0.5) +
  geom_smooth(formula=y ~ x, method="lm") +
  ggtitle("Scatterplots") +
  facet_wrap(~ key, scales="free_x")
```



The scatterplots indicate that most variables are upward sloping in `price`. We can see large variances for most features, especially at large values. `sqft_living` (and `sqft_living15`) have some of the most straightforward trends of any of the variables. `bedrooms`, `bathrooms`, and `grade` demonstrate a similar strong trend, but with a wider spread. These trends comport with basic economic theory. It is harder to draw conclusions from the indicator and limited selection variables like `condition`, `floors`, `view`, `waterfront`, `spr_sum`, and `yr_sold`, but generally they seem to have positive effects. Finally, `lat` and `long` demonstrate extreme variance at some particular values, likely near Seattle.

2. Naive Multiple Regression

```
model.base <- lm(price ~ ., data=kc_house)
summary(model.base)
```

```
##
## Call:
## lm(formula = price ~ ., data = kc_house)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1257365   -99204    -9649    76087  4349543
##
## Coefficients: (1 not defined because of singularities)
##                 Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   -2.452e+05  2.058e+04 -11.916  < 2e-16 ***
## bedrooms      -3.446e+04  1.899e+03 -18.152  < 2e-16 ***
## bathrooms      4.241e+04  3.277e+03  12.944  < 2e-16 ***
## sqft_living    1.470e+02  4.402e+00  33.397  < 2e-16 ***
## sqft_lot       1.267e-01  4.816e-02   2.630 0.008546 **
## floors         9.639e+02  3.604e+03   0.267 0.789105
## waterfront     5.922e+05  1.742e+04  33.999  < 2e-16 ***
## view           4.935e+04  2.141e+03  23.052  < 2e-16 ***
## condition      3.201e+04  2.360e+03  13.562  < 2e-16 ***
## grade          9.710e+04  2.164e+03  44.862  < 2e-16 ***
## sqft_above     3.328e+01  4.382e+00   7.596 3.18e-14 ***
## sqft_basement         NA         NA      NA       NA
## yr_built      -3.078e+03  1.167e+02 -26.373  < 2e-16 ***
## yr_renovated   6.213e+02  1.215e+02   5.114 3.18e-07 ***
## lat            5.631e+05  1.053e+04  53.471  < 2e-16 ***
## long           1.177e+05  1.198e+04   9.827  < 2e-16 ***
## sqft_living15  2.733e+01  3.451e+00   7.920 2.48e-15 ***
## sqft_lot15    -3.920e-01  7.362e-02  -5.325 1.02e-07 ***
## yr_sold        2.885e+04  2.972e+03   9.707  < 2e-16 ***
## spr_sum        1.069e+04  2.826e+03   3.783 0.000156 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 202200 on 21594 degrees of freedom
## Multiple R-squared:  0.6969, Adjusted R-squared:  0.6967
## F-statistic:  2759 on 18 and 21594 DF,  p-value: < 2.2e-16
```

Most of the coefficients are highly statistically significant at the 5% level, except for `floors`. Of note is that `sqft_basement` has no estimated coefficient. This is because of collinearity with `sqft_lot` and `sqft_above`, as the warning message about singularities indicates.

2.1 Economic Interpretation

The economic interpretation of several of the coefficients raises some questions about our baseline model. The model estimates an intercept on the order of -$245k, which obviously makes no sense on its own. However, the positive coefficients of a similar magnitude for `lat` and `long` counterbalance this, which indicates that houses in the western and northern parts of the county (near major metropolitan areas) are more expensive.

Turning to the more economically relevant variables, we can see that most of the predictors have positive coefficients. This accords with common sense. Examining each predictor in turn:

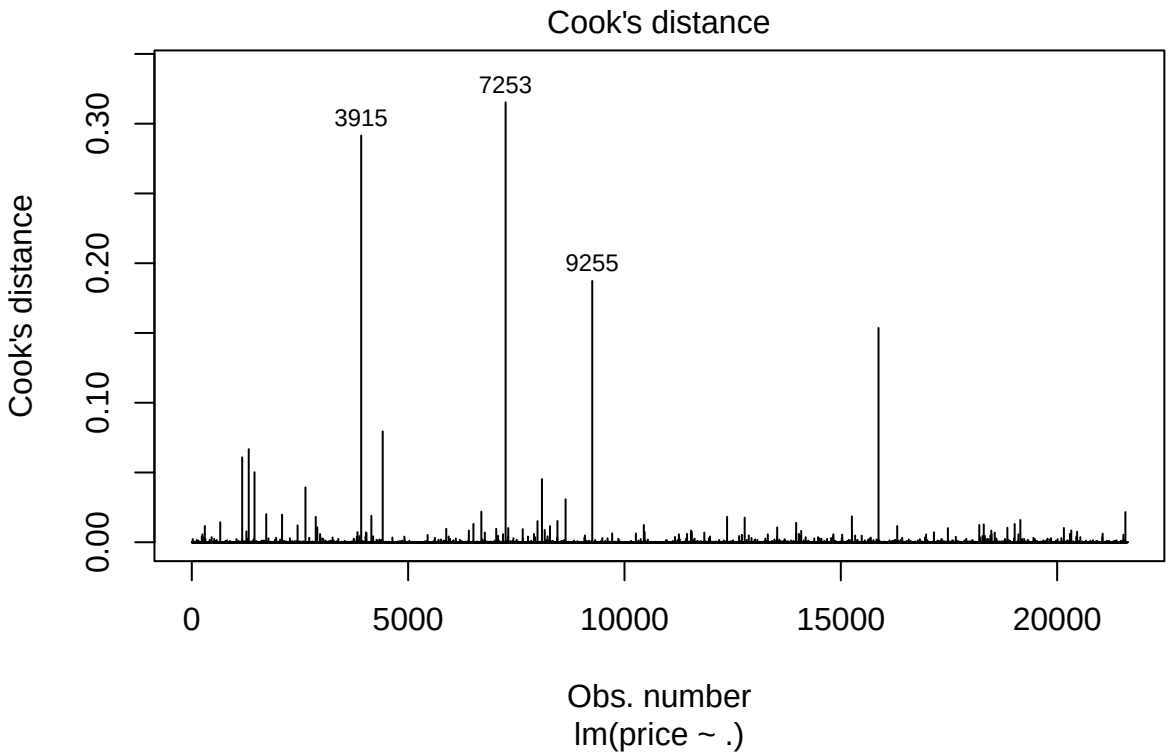
- `bedrooms` has a large negative coefficient. Holding all other variables (especially square footage) constant, this could make sense, as more bedrooms in the same size house would mean smaller, less desirable rooms, but the magnitude of the effect is quite large (about $34k per bedroom) compared to the area value of an additional typical 10' x 12' bedroom [120 sqft times a coefficient of $150 (or $180 when including the effect of `sqft_above`)], which is about $20k. This negative coefficient seems economically unreasonable.
- `bathrooms` has a large positive coefficient, which would make economic sense. More bathrooms would make a house more desirable while taking up less square footage than a bedroom. The size of the effect is large, at over $40k.
- `sqft_living` and `sqft_lot` have positive coefficients, although the scale of the coefficients makes it clear that square footage of living space is much more important than property space. The coefficient of `sqft_living` is about $150 per square foot, which is eminently reasonable for residential properties.

- `floors` has a moderate positive coefficient. Larger houses would tend to have more floors and be worth more, so this makes sense, though the magnitude is slight.
- `waterfront`, `view`, `condition`, and `grade` all have moderate to large positive coefficients. All of these accord with economic intuition, as proximity to the water, view, and quality of construction should all have a positive effect on home value. It is possible that a better view could be associated with a nearly $50k difference in home value (though `view` can range up to 4). The scale of `waterfront` is quite large, at over half a million dollars (since `waterfront` is an indicator variable). This isn't necessarily incorrect; it may be that especially large, expensive houses happen to be built near the water.
- `yr_built` has a negative coefficient. This implies that (all else equal) newer houses are less valuable than older houses, which does not make sense. `yr_renovated` has a positive effect, which aligns with economic intuition: a more recently renovated house should be more valuable. The scale of the coefficient of `yr_renovated` seems appropriate for the size of the effect.
- `sqft_living15` has a positive coefficient, while `sqft_lot15` has a negative one (of a magnitude less than 1.0). While these factors do seem to affect house prices, they do not appear to be the primary effects. This is reasonable, as they are somewhat of a proxy for the surrounding neighborhood. While this has some impact on the value of a home, we would expect it to be less than the properties of the home itself. Larger surrounding houses are associated with somewhat higher values, while larger surrounding lots are associated with lower values. This may be because smaller lots are more common in the most desirable urban areas.
- The two indicator variables we engineered, `yr_sold` and `spr_sum`, both have positive coefficients on the order of tens of thousands of dollars. This indicates that homes sold during 2015 and homes sold during the spring/summer were marginally more expensive than those sold in 2014 or during the fall/winter. These economic interpretations are eminently reasonable.

3. Outlier Detection

We first examine data points with a high leverage according to our baseline model.

```
plot(model.base, which=4)
```




```
short_feats <- c("price", "bedrooms", "bathrooms", "sqft_living", "sqft_lot",
                 "floors", "yr_built")
kc_house[c("3915", "7253", "9255"), short_feats]
```

```
##        price bedrooms bathrooms sqft_living sqft_lot floors yr_built
## 3915 7062500        5      4.50       10040    37325    2.0       40
## 7253 7700000        6      8.00       12050    27600    2.5       10
## 9255 6885000        6      7.75        9890    31374    2.0      101
```

It appears that the highest-leverage examples are the three houses with the highest prices in the dataset. However, looking at the data gives no indications of being an outlier on any of the predictor variables. As such, it appears that these observations are just high variance at the far right tail of the price distribution (and we can't remove them just for having large residuals), so we opt to leave them in our dataset.

Based on the five number summaries, we can examine some other potential outliers. We start with the extremely high maximum value of `bedrooms`.

```
kc_house[kc_house$bedrooms > 8, short_feats]
```

```
##         price bedrooms bathrooms sqft_living sqft_lot floors yr_built
## 4097   599999        9      4.50        3830     6988    2.5       38
## 4236   700000        9      3.00        3680     4400    2.0        8
## 6080  1280000        9      4.50        3650     5000    2.0       15
## 8547   450000        9      7.50        4050     6504    2.0       96
## 8758   520000       11      3.00        3000     4960    2.0       18
```

```
## 13315 1148000       10      5.25        4590    10920    1.0      108
## 15162  650000       10      2.00        3610    11914    2.0       58
## 15871  640000       33      1.75        1620     6000    1.0       47
## 16845 1400000        9      4.00        4620     5508    2.5       15
## 18444  934000        9      3.00        2820     4480    2.0       18
## 19255  660000       10      3.00        2920     3745    2.0       13
```

```
kc_house[kc_house$bedrooms == 33, "bedrooms"] <- 3 # Impute correct value
```

The 11-bedroom house seems small, but not unreasonable. The 33-bedroom house, on the other hand, seems to be very out of place. This is likely the result of a data entry mistake. A 1600 sqft, single-story house built in 1947 probably has 3 bedrooms, not 33. Rather than dropping this observation, we simply correct it.

We now look at potential outliers with no bedrooms or bathrooms.

```
kc_house[kc_house$bedrooms == 0 | kc_house$bathrooms == 0, short_feats]
```

```
##         price bedrooms bathrooms sqft_living sqft_lot floors yr_built
## 876   1095000        0      0.00        3064     4764    3.5       90
## 1150    75000        1      0.00         670    43377    1.0       66
## 3120   380000        0      0.00        1470      979    3.0      106
## 3468   288000        0      1.50        1430     1650    3.0       99
## 4869   228000        0      1.00         390     5900    1.0       53
## 5833   280000        1      0.00         600    24501    1.0       50
## 6995  1295650        0      0.00        4810    28008    2.0       90
## 8478   339950        0      2.50        2290     8319    2.0       85
## 8485   240000        0      2.50        1810     5669    2.0      103
## 9774   355000        0      0.00        2460     8049    2.0       90
## 9855   235000        0      0.00        1470     4800    2.0       96
## 10482  484000        1      0.00         690    23244    1.0       48
## 12654  320000        0      2.50        1490     7111    2.0       99
## 14424  139950        0      0.00         844     4269    1.0       13
## 18380  265000        0      0.75         384   213444    1.0      103
## 19453  142000        0      0.00         290    20875    1.0       63
```

Some of these rows look like they are missing data (e.g. homes with 1400-4800 square feet of living space). Others are more unusual, with less than 1000 square feet of living space on very large (tens of thousands of square feet) lots. We are unsure of the nature of these "houses", but a tiny building without basic necessities on very large lots are definitely outliers in a model for house prices. Since there are only 16 total rows with zero values, we decide to exclude these rows as outliers.

```
kc_house <- kc_house[kc_house$bedrooms != 0 & kc_house$bathrooms != 0,]
```

Next, let's examine lot square footage. From our descriptive analysis, we know that the data is far from normally distributed with an extremely long right tail. However, we would still like to have some metric to identify outliers in the dataset. To be conservative, we examine data which is six or more standard deviations above the mean for `sqft_lot`.

```
lot_upper <- mean(kc_house$sqft_lot) + 6 * sd(kc_house$sqft_lot)
short_feats <- c("price", "bedrooms", "bathrooms", "sqft_living", "sqft_lot",
                 "floors", "sqft_lot15")
large_lots <- kc_house[kc_house$sqft_lot > lot_upper, short_feats]
large_lots <- large_lots[order(-large_lots$sqft_lot), ] # Sort descending
head(large_lots, n=20)
```

```
##         price bedrooms bathrooms sqft_living sqft_lot floors sqft_lot15
## 1720   700000        4      1.00        1300  1651359    1.0     425581
## 17320  190000        2      1.00         710  1164794    1.0      16730
## 7648   542500        5      3.25        3010  1074218    1.5      68825
## 7770   855000        4      3.50        4030  1024068    2.0      11700
## 3950   998000        4      3.25        3770   982998    2.0      37141
## 4442   790000        2      3.00        2560   982278    1.0      40946
## 6692  1998000        2      2.50        3900   920423    2.0     411962
## 7078  1650000        4      3.25        3920   881654    3.0     112384
## 9715   937500        4      4.00        5545   871200    2.0     871200
## 20453 1600000        4      5.50        6530   871200    2.0     858132
## 4541   550000        3      2.00        3650   843309    2.0     273992
## 13007  750000        3      2.50        2350   715690    1.5     325393
## 13478  849900        2      2.00        2280   641203    2.0     224334
## 16189  700000        3      2.50        2530   623779    1.0     100623
## 7295   700000        2      1.75        1679   577605    2.0     358934
## 17826  425000        3      2.75        1360   542322    1.0      60548
## 17577  375000        1      1.00         800   533610    1.5     216057
## 12920  428000        3      1.75        1580   507038    1.0     210394
## 2965   999000        3      2.75        2830   505166    1.0      21988
## 20422  920000        4      3.75        4030   503989    2.0      71874
```

```
nrow(large_lots)
```

```
## [1] 93
```

```
large_lots[large_lots$bedrooms == 1, ]
```

```
##        price bedrooms bathrooms sqft_living sqft_lot floors sqft_lot15
## 17577 375000        1         1         800   533610    1.5     216057
```

We can see that there are 93 observations with lot sizes more than 6 standard deviations above the mean. This is a lot of observations to exclude as outliers, especially when we can see from the top 20 observations that the distribution is relatively smooth.

However, the largest property is nearly 50% larger than the second-largest, which is a large enough difference to be reasonably considered an outlier. The second largest observation is one of the smallest houses in the dataset (710 square feet) on the second-largest property (1.1M sqft), so we also consider this an outlier when compared to the other properties with large lots. Similarly, we exclude the only single-bedroom house on a large lot. The rest of the large lot properties occur frequently enough that we will not consider them outliers.

```
# Exclude exceptionally large lots
kc_house <- kc_house[kc_house$sqft_lot < 1100000, ]
# Exclude small houses on large lots
kc_house <- kc_house[kc_house$sqft_lot < lot_upper | kc_house$bedrooms > 1, ]
```

Finally, we examine the tail end of `sqft_lot15`.

```
head(kc_house[order(-kc_house$sqft_lot15), short_feats], n=15)
```

```
##         price bedrooms bathrooms sqft_living sqft_lot floors sqft_lot15
## 9715   937500        4      4.00        5545   871200    2.0     871200
```

```
## 20453 1600000        4      5.50        6530   871200    2.0     858132
## 13465  790000        3      2.50        2640   432036    1.5     560617
## 8665   549950        3      1.75        2930   266587    2.0     438213
## 3802   637000        4      3.50        3080   118918    2.0     434728
## 19157  858000        4      3.50        4370   422967    1.0     422967
## 6692  1998000        2      2.50        3900   920423    2.0     411962
## 15621  180000        2      1.00         960    87991    1.5     392040
## 21432  800000        4      3.25        3540   159430    2.0     392040
## 17660  370000        2      1.00        2360   105850    1.0     386812
## 11184  319000        3      1.75        1640    53400    1.0     380279
## 11565  350000        3      1.75        1680   250470    1.0     360000
## 7295   700000        2      1.75        1679   577605    2.0     358934
## 3537   430000        3      1.50        1810   349351    1.5     339332
## 15558  230000        3      1.00        1120    32250    1.0     335289
```

This distribution seems fairly continuous until the last three points, when it suddenly jumps to over 550k and then to over 850k. These values are far in excess of the rest of the distribution, so we exclude them as outliers.

```
# Exclude exceptionally neighboring large lots
kc_house <- kc_house[kc_house$sqft_lot15 < 550000, ]
```

```
summary_df <- sapply(kc_house, fivenum)
rownames(summary_df) <- c("Min", "Q1", "Median", "Q3", "Max")
summary_df <- t(summary_df) # Transpose to fit on page
stargazer(summary_df, type="text")
```

New Summary Statistics

```
##
## ========================================================
##                 Min     Q1    Median    Q3       Max
## --------------------------------------------------------
## price         78,000 322,000 450,000 645,000 7,700,000
## bedrooms        1       3       3       4       11
## bathrooms     0.500   1.750   2.250   2.500      8
## sqft_living    370    1,430   1,910   2,550   13,540
## sqft_lot       520    5,040   7,617  10,676  1,074,218
## floors          1       1     1.500     2      3.500
## waterfront      0       0       0       0        1
## view            0       0       0       0        4
## condition       -2      0       0       1        2
## grade           -4      0       0       1        6
## sqft_above     370    1,190   1,560   2,210    9,410
## sqft_basement   0       0       0      560     4,820
## yr_built        0      51      75      97       115
## yr_renovated    0      54      77      99       115
## lat           0.156   0.471   0.572   0.678    0.778
## long          0.315   1.125   1.231   1.328    1.519
## sqft_living15  399    1,490   1,840   2,360    6,210
```

```
## sqft_lot15     651    5,100   7,620  10,080   438,213
## yr_sold          0        0       0       1         1
## spr_sum          0        0       1       1         1
## ---------------------------------------------------------
```

Our new summary statistics, after removing outliers, look more reasonable than the set on the raw data. We can see that some of the concerns about outlier values that we originally had have been resolved.

3.1 Re-estimation

```
model.rebase <- lm(price ~ ., data=kc_house)
summary(model.rebase)
```

```
##
## Call:
## lm(formula = price ~ ., data = kc_house)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1257853   -99207    -9342    76079  4333190
##
## Coefficients: (1 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.369e+05  2.066e+04 -11.465  < 2e-16 ***
## bedrooms      -3.754e+04  1.986e+03 -18.904  < 2e-16 ***
## bathrooms      4.363e+04  3.292e+03  13.254  < 2e-16 ***
## sqft_living    1.488e+02  4.422e+00  33.646  < 2e-16 ***
## sqft_lot       9.806e-02  5.257e-02   1.865 0.062148 .
## floors         5.031e+02  3.607e+03   0.139 0.889078
## waterfront     5.917e+05  1.740e+04  34.008  < 2e-16 ***
## view           4.874e+04  2.141e+03  22.769  < 2e-16 ***
## condition      3.224e+04  2.359e+03  13.669  < 2e-16 ***
## grade          9.774e+04  2.173e+03  44.988  < 2e-16 ***
## sqft_above     3.329e+01  4.381e+00   7.599 3.11e-14 ***
## sqft_basement         NA         NA      NA       NA
## yr_built      -3.086e+03  1.166e+02 -26.479  < 2e-16 ***
## yr_renovated   6.127e+02  1.213e+02   5.049 4.47e-07 ***
## lat            5.615e+05  1.053e+04  53.340  < 2e-16 ***
## long           1.173e+05  1.200e+04   9.775  < 2e-16 ***
## sqft_living15  2.666e+01  3.461e+00   7.703 1.39e-14 ***
## sqft_lot15    -3.899e-01  7.861e-02  -4.960 7.09e-07 ***
## yr_sold        2.922e+04  2.969e+03   9.839  < 2e-16 ***
## spr_sum        1.084e+04  2.823e+03   3.840 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 201900 on 21572 degrees of freedom
## Multiple R-squared:  0.6978, Adjusted R-squared:  0.6976
## F-statistic:  2768 on 18 and 21572 DF,  p-value: < 2.2e-16
```

Most of the coefficient estimates and their standard errors are very close to the values from the original model. The multiple R^2 and adjusted R^2 are very slightly higher, but hardly enough to claim any practical

difference. The only major difference is that the coefficient of `sqft_lot` is no longer significant at the 5% level.

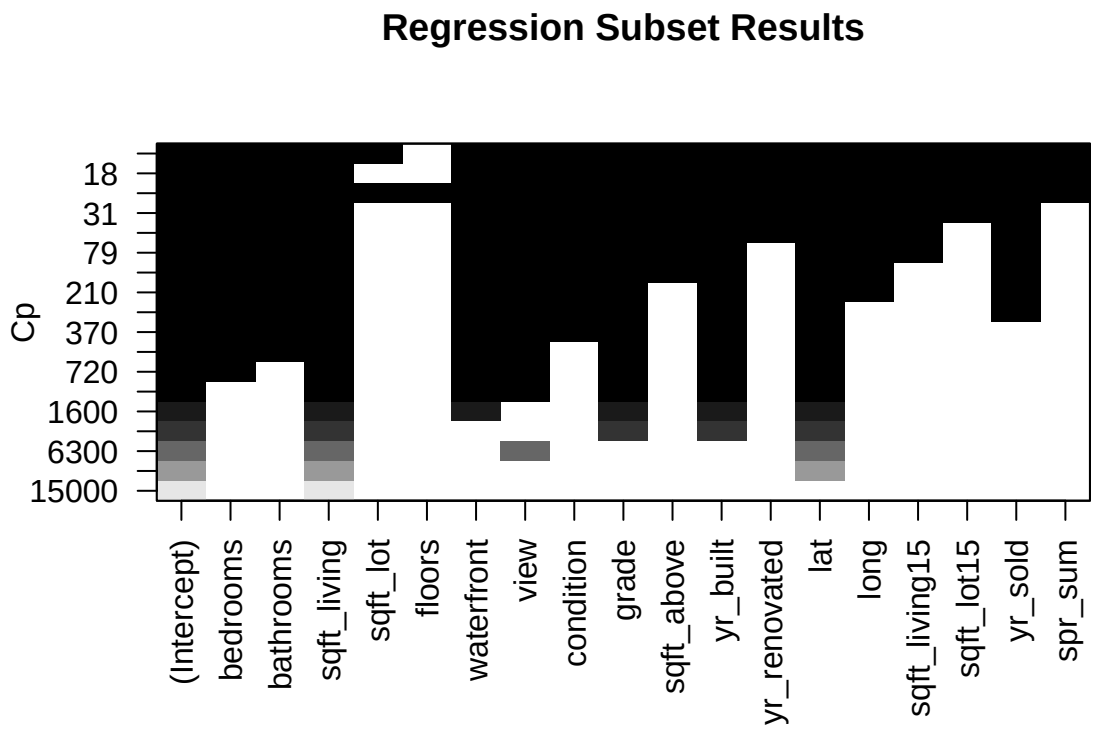
4. Feature Selection

At this time, we will also remove the `sqft_basement` feature before proceeding to feature selection, since we know it is perfectly collinear with other features in the dataset.

```
kc_house <- subset(kc_house, select=-c(sqft_basement))
```

4.1 Mallows's Cp

```
ss <- regsubsets(price ~ ., method=c("exhaustive"), nbest=1, nvmax=18, data=kc_house)
plot(ss, scale="Cp", main="Regression Subset Results")
```



Based on the plot, it appears that the intercept, `sqft_living`, and `lat` are the most important variables. The least important variables appear to be `sqft_lot` and `floors`. This makes sense, as those coefficients were not even statistically significant in our re-estimated regression.

We decline to plot the Cp values against each other since the labels would be unmanageable with 10+ variables. However, we can examine the numerical values of Cp:

```
ss.sum <- summary(ss)
cp.df <- data.frame(ss.sum$which)
cp.df$cp <- ss.sum$cp
cp.df
```

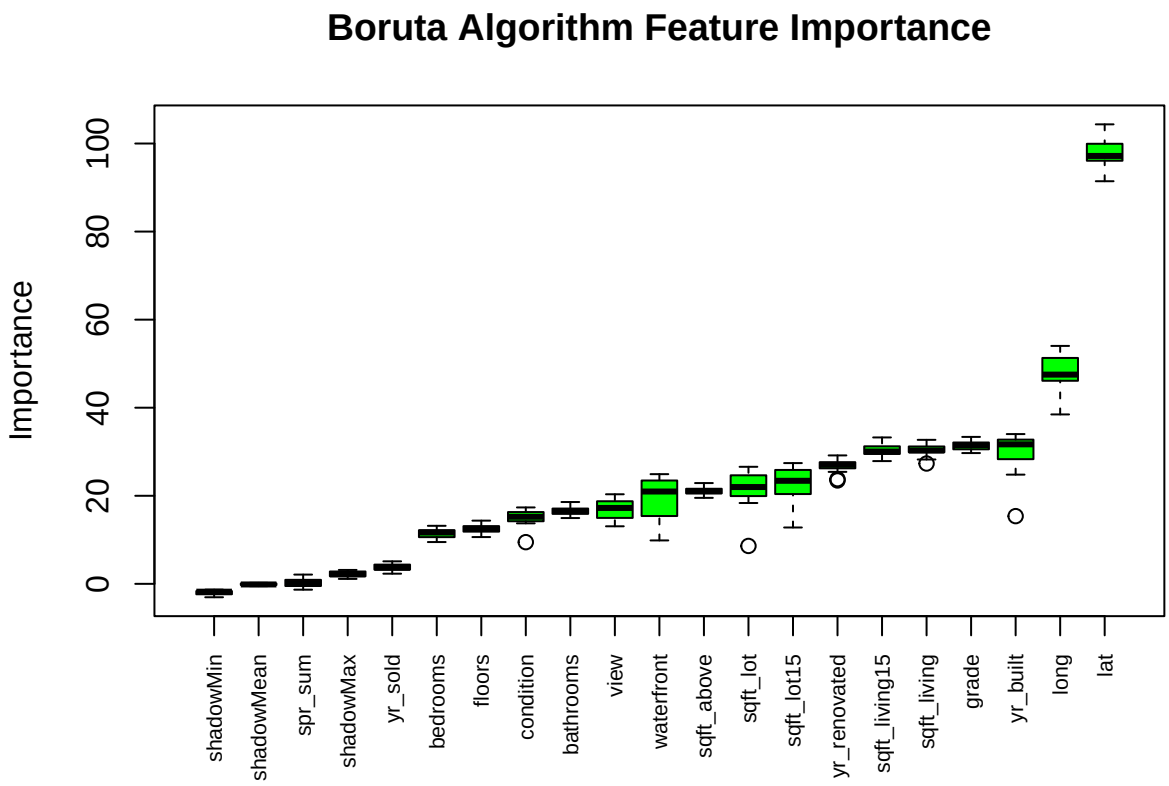
```
##    X.Intercept. bedrooms bathrooms sqft_living sqft_lot floors waterfront  view
## 1          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE      FALSE FALSE
## 2          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE      FALSE FALSE
## 3          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE      FALSE  TRUE
## 4          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE      FALSE FALSE
## 5          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE       TRUE FALSE
## 6          TRUE    FALSE     FALSE        TRUE    FALSE  FALSE       TRUE  TRUE
## 7          TRUE     TRUE     FALSE        TRUE    FALSE  FALSE       TRUE  TRUE
## 8          TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 9          TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 10         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 11         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 12         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 13         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 14         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 15         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 16         TRUE     TRUE      TRUE        TRUE    FALSE  FALSE       TRUE  TRUE
## 17         TRUE     TRUE      TRUE        TRUE     TRUE  FALSE       TRUE  TRUE
## 18         TRUE     TRUE      TRUE        TRUE     TRUE   TRUE       TRUE  TRUE
##    condition grade sqft_above yr_built yr_renovated   lat  long sqft_living15
## 1      FALSE FALSE      FALSE    FALSE        FALSE FALSE FALSE         FALSE
## 2      FALSE FALSE      FALSE    FALSE        FALSE  TRUE FALSE         FALSE
## 3      FALSE FALSE      FALSE    FALSE        FALSE  TRUE FALSE         FALSE
## 4      FALSE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 5      FALSE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 6      FALSE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 7      FALSE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 8      FALSE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 9       TRUE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 10      TRUE  TRUE      FALSE     TRUE        FALSE  TRUE FALSE         FALSE
## 11      TRUE  TRUE      FALSE     TRUE        FALSE  TRUE  TRUE         FALSE
## 12      TRUE  TRUE       TRUE     TRUE        FALSE  TRUE  TRUE         FALSE
## 13      TRUE  TRUE       TRUE     TRUE        FALSE  TRUE  TRUE          TRUE
## 14      TRUE  TRUE       TRUE     TRUE         TRUE  TRUE  TRUE          TRUE
## 15      TRUE  TRUE       TRUE     TRUE         TRUE  TRUE  TRUE          TRUE
## 16      TRUE  TRUE       TRUE     TRUE         TRUE  TRUE  TRUE          TRUE
## 17      TRUE  TRUE       TRUE     TRUE         TRUE  TRUE  TRUE          TRUE
## 18      TRUE  TRUE       TRUE     TRUE         TRUE  TRUE  TRUE          TRUE
##    sqft_lot15 yr_sold spr_sum          cp
## 1       FALSE   FALSE   FALSE 14636.25931
## 2       FALSE   FALSE   FALSE  9398.01076
## 3       FALSE   FALSE   FALSE  6305.58996
## 4       FALSE   FALSE   FALSE  4023.09990
## 5       FALSE   FALSE   FALSE  1628.51445
## 6       FALSE   FALSE   FALSE   958.75594
## 7       FALSE   FALSE   FALSE   720.08199
## 8       FALSE   FALSE   FALSE   490.45014
## 9       FALSE   FALSE   FALSE   371.23347
```

```
## 10      FALSE    TRUE   FALSE   268.10381
## 11      FALSE    TRUE   FALSE   212.41385
## 12      FALSE    TRUE   FALSE   128.41336
## 13      FALSE    TRUE   FALSE    78.70563
## 14      FALSE    TRUE   FALSE    54.49527
## 15       TRUE    TRUE   FALSE    30.95892
## 16       TRUE    TRUE    TRUE    18.48670
## 17       TRUE    TRUE    TRUE    17.01945
## 18       TRUE    TRUE    TRUE    19.00000
```

It is clear from the Cp values that the best model includes all of the variables except for `floors`. The second best model, with 16 features, drops both `floors` and `sqft_lot`. Both of these models outperform the full model with all 18 features on Mallows's Cp.

4.2 Boruta Algorithm

```
Bor.res <- Boruta(price ~ ., data=kc_house)
# Plot results of Boruta test with all labels
plot(Bor.res, xlab = "", xaxt = "n", main = "Boruta Algorithm Feature Importance")
lz <- lapply(1:ncol(Bor.res$ImpHistory), function(i)
  Bor.res$ImpHistory[is.finite(Bor.res$ImpHistory[ ,i]), i])
names(lz) <- colnames(Bor.res$ImpHistory)
Bor.labels <- sort(sapply(lz, median))
axis(side=1, las=2, labels=names(Bor.labels), at=1:ncol(Bor.res$ImpHistory), cex.axis=0.7)
```



From the graph of the Boruta feature importance, it appears that the only rejected value is `spr_sum`.

```
boruta_signif_Conf <- names(Bor.res$finalDecision[Bor.res$finalDecision %in%
    c("Confirmed")])
boruta_signif_Tent <- names(Bor.res$finalDecision[Bor.res$finalDecision %in%
    c("Tentative")])
boruta_signif_Reject <- names(Bor.res$finalDecision[Bor.res$finalDecision %in%
    c("Rejected")])
print(boruta_signif_Conf)
```

```
##  [1] "bedrooms"      "bathrooms"     "sqft_living"   "sqft_lot"
##  [5] "floors"        "waterfront"    "view"          "condition"
##  [9] "grade"         "sqft_above"    "yr_built"      "yr_renovated"
## [13] "lat"           "long"          "sqft_living15" "sqft_lot15"
## [17] "yr_sold"
```

```
print(boruta_signif_Tent)
```

```
## character(0)
```

```
print(boruta_signif_Reject)
```

```
## [1] "spr_sum"
```

```
# Rank the best predictors according to Boruta
attStats(Bor.res)[order(-attStats(Bor.res)$meanImp), ]
```

```
##                  meanImp  medianImp    minImp     maxImp   normHits  decision
## lat           97.9826988 97.1755863 91.434446 104.351437 1.00000000 Confirmed
## long          48.0992356 47.5189879 38.475134  54.036028 1.00000000 Confirmed
## grade         31.3905449 31.5035596 29.703602  33.371420 1.00000000 Confirmed
## sqft_living15 30.3946812 30.0470151 27.876131  33.249003 1.00000000 Confirmed
## sqft_living   30.2544787 30.2672739 27.327747  32.706338 1.00000000 Confirmed
## yr_built      29.7508994 31.6804996 15.377952  34.013108 1.00000000 Confirmed
## yr_renovated  26.8137128 27.0492396 23.482429  29.168258 1.00000000 Confirmed
## sqft_lot15    22.7156769 23.4016466 12.798313  27.426704 1.00000000 Confirmed
## sqft_lot      21.6284262 21.9865226  8.606753  26.581622 1.00000000 Confirmed
## sqft_above    21.0557038 21.0342053 19.525556  22.890263 1.00000000 Confirmed
## waterfront    19.4288937 20.9688569  9.859009  24.909180 1.00000000 Confirmed
## view          16.8627912 17.2372324 13.073664  20.344300 1.00000000 Confirmed
## bathrooms     16.6379310 16.5274008 14.943533  18.579915 1.00000000 Confirmed
## condition     15.0391041 15.2358712  9.470778  17.354623 1.00000000 Confirmed
## floors        12.4513033 12.4936177 10.626645  14.354465 1.00000000 Confirmed
## bedrooms      11.4647930 11.6533941  9.495038  13.178073 1.00000000 Confirmed
## yr_sold        3.7625518  3.6547738  2.310987   5.113078 1.00000000 Confirmed
## spr_sum        0.2398246  0.1645428 -1.324477   2.102274 0.06666667  Rejected
```

The Boruta test identifies `lat` and `long` as the most important features by far. The old adage about location is evidently correct! These are followed by `grade` and `sqft_living`, which makes sense.

4.3 Re-estimation

In light of the results of our feature selection tests and the significance tests from the original regression, we decided to drop `sqft_lot`, `floors`, and `spr_sum`. We are left with 15 remaining features: `bedrooms`, `bathrooms`, `sqft_living`, `waterfront`, `view`, `condition`, `grade`, `sqft_above`, `yr_built`, `yr_renovated`, `lat`, `long`, `sqft_living15`, `sqft_lot15`, and `yr_sold`.

```
model.featsel <- lm(price ~ . -sqft_lot -floors -spr_sum, data=kc_house)
summary(model.featsel)
```

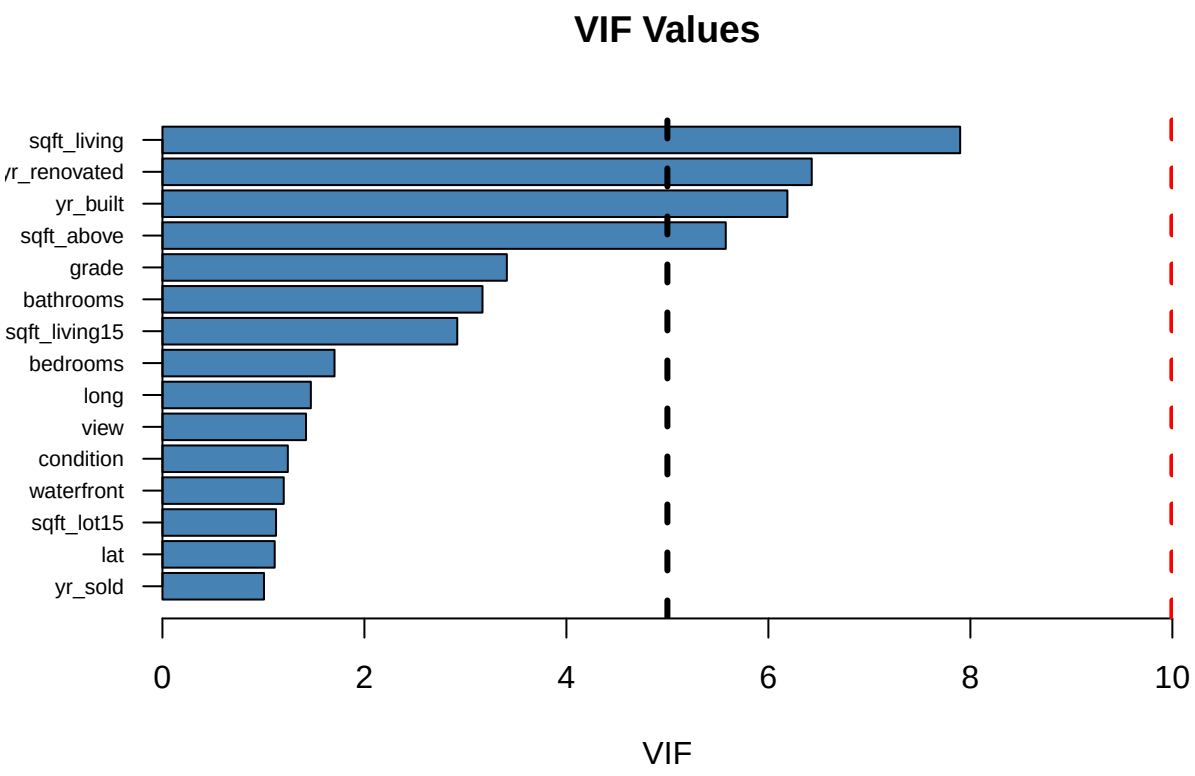
```
##
## Call:
## lm(formula = price ~ . - sqft_lot - floors - spr_sum, data = kc_house)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1248217   -99151    -9484    76605  4325082
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.274e+05  2.052e+04 -11.085  < 2e-16 ***
## bedrooms      -3.766e+04  1.985e+03 -18.976  < 2e-16 ***
## bathrooms      4.376e+04  3.184e+03  13.747  < 2e-16 ***
## sqft_living    1.488e+02  4.211e+00  35.343  < 2e-16 ***
## waterfront     5.909e+05  1.740e+04  33.957  < 2e-16 ***
## view           4.879e+04  2.139e+03  22.806  < 2e-16 ***
## condition      3.233e+04  2.354e+03  13.734  < 2e-16 ***
## grade          9.800e+04  2.165e+03  45.275  < 2e-16 ***
## sqft_above     3.360e+01  3.925e+00   8.561  < 2e-16 ***
## yr_built      -3.094e+03  1.164e+02 -26.585  < 2e-16 ***
## yr_renovated   6.177e+02  1.210e+02   5.106 3.32e-07 ***
## lat            5.616e+05  1.046e+04  53.680  < 2e-16 ***
## long           1.153e+05  1.184e+04   9.735  < 2e-16 ***
## sqft_living15  2.632e+01  3.427e+00   7.681 1.65e-14 ***
## sqft_lot15    -2.881e-01  5.704e-02  -5.052 4.42e-07 ***
## yr_sold        3.064e+04  2.947e+03  10.395  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 201900 on 21575 degrees of freedom
## Multiple R-squared:  0.6976, Adjusted R-squared:  0.6974
## F-statistic:  3318 on 15 and 21575 DF,  p-value: < 2.2e-16
```

Our new model is extremely similar to the previous model, but it is more parsimonious. The economic interpretations are unchanged.

5. Multicollinearity

```
vif_values <- sort(vif(model.featsel))
# Plot the sorted values
```

```
pos <- barplot(vif_values, main="VIF Values", horiz=TRUE, col="steelblue", yaxt="n",
               ylab="", xlab="VIF", xlim=c(0, 10))
axis(side=2, labels=names(vif_values), at=pos, las=1, cex.axis=0.7)
abline(v=5, lwd=3, lty=2)
abline(v=10, lwd=3, lty=2, col="red")
```



We can see high collinearity in four columns. Common sense would tell us that `sqft_living` and `sqft_above` are collinear, and `yr_renovated` and `yr_built`. We will try dropping `sqft_above` and `yr_renovated`, since our feature selection tests indicated that these features were much less important than `sqft_living` and `yr_built`.

5.1 Re-estimation

```
model.noncol <- lm(price ~ . -sqft_lot -floors -spr_sum -sqft_above -yr_renovated,
                   data=kc_house)
summary(model.noncol)
```

```
##
## Call:
## lm(formula = price ~ . - sqft_lot - floors - spr_sum - sqft_above -
##     yr_renovated, data = kc_house)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1220721   -99863   -10110    77239  4359049
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.795e+05  1.998e+04  -8.984  < 2e-16 ***
## bedrooms      -3.843e+04  1.987e+03 -19.336  < 2e-16 ***
## bathrooms      4.549e+04  3.138e+03  14.497  < 2e-16 ***
## sqft_living    1.702e+02  3.379e+00  50.363  < 2e-16 ***
## waterfront     5.999e+05  1.742e+04  34.441  < 2e-16 ***
## view           4.586e+04  2.113e+03  21.708  < 2e-16 ***
## condition      2.745e+04  2.295e+03  11.963  < 2e-16 ***
## grade          1.025e+05  2.119e+03  48.382  < 2e-16 ***
## yr_built      -2.557e+03  6.704e+01 -38.149  < 2e-16 ***
## lat            5.510e+05  1.043e+04  52.823  < 2e-16 ***
## long           9.547e+04  1.170e+04   8.162 3.47e-16 ***
## sqft_living15  2.811e+01  3.408e+00   8.249  < 2e-16 ***
## sqft_lot15    -2.777e-01  5.715e-02  -4.859 1.19e-06 ***
## yr_sold        3.016e+04  2.953e+03  10.213  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 202400 on 21577 degrees of freedom
## Multiple R-squared:  0.6962, Adjusted R-squared:  0.696
## F-statistic:  3803 on 13 and 21577 DF,  p-value: < 2.2e-16
```

Our model maintains a very similar predictive power as the previous two models, but now with only 13 features down from the original 19. All of the coefficients are extremely statistically significant at the 5% level.

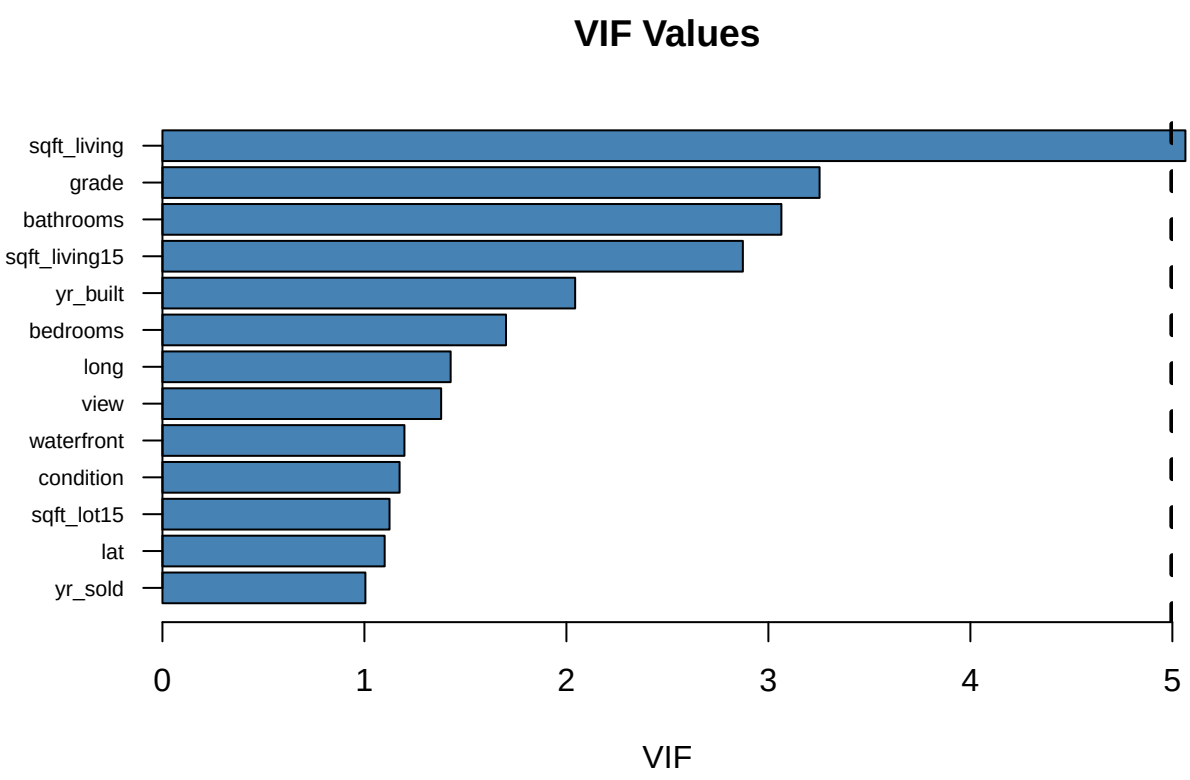
Economic Interpretation

The economic interpretation of some of our variables has not changed, but the values of the coefficients have slightly.

- The intercept is less negative by about $40k.
- `sqft_living` has assumed most of the coefficient of `sqft_above`.
- The coefficient of `condition` has slightly decreased and `grade` has increased.
- `yr_built` is still negative, but much less so. It appears that without the collinear column `yr_renovated`, the coefficient has moved more in line with our economic intuition that newer houses should be worth more.

Collinearity

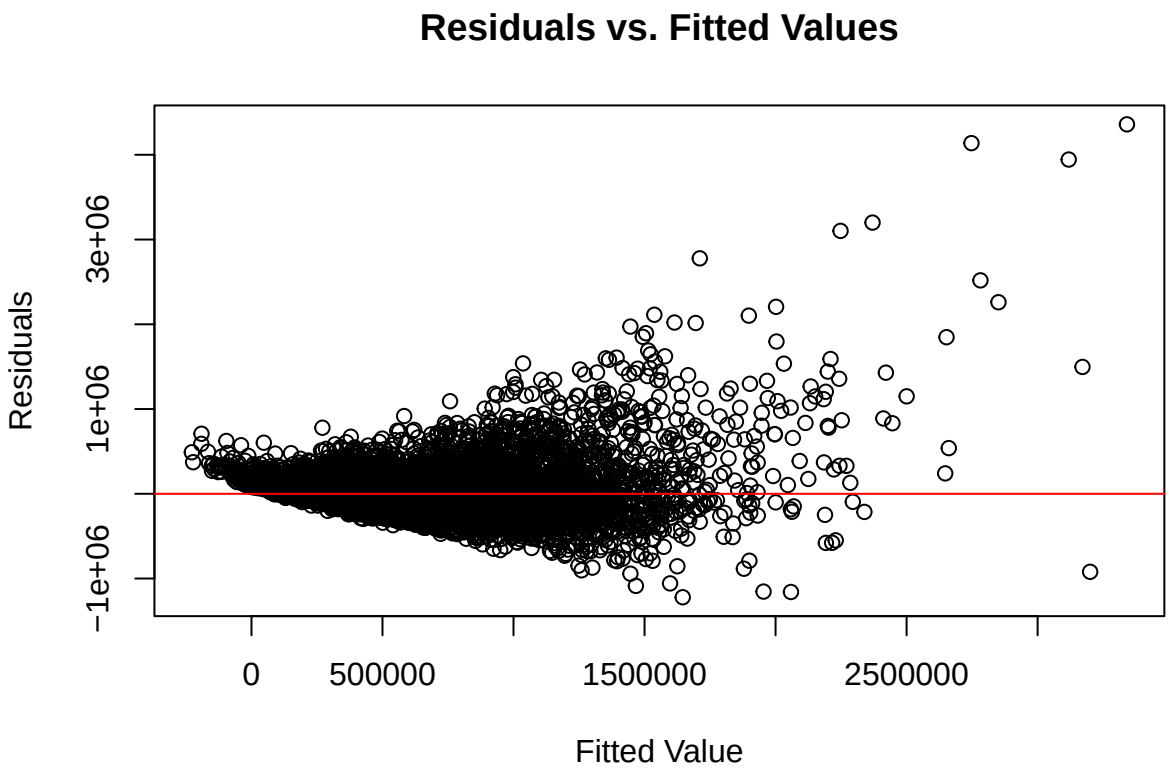
```r
vif_values <- sort(vif(model.noncol))
# Plot the sorted values
pos <- barplot(vif_values, main="VIF Values", horiz=TRUE, col="steelblue", yaxt="n",
               ylab="", xlab="VIF", xlim=c(0, 5))
axis(side=2, labels=names(vif_values), at=pos, las=1, cex.axis=0.7)
abline(v=5, lwd=3, lty=2)
```



Checking our new model for collinearity shows `sqft_living` is still above 5. It is likely collinear with `sqft_living15`. However, since the value is just barely above 5 (and we have already removed 2 features), we elect to leave both features in our regression. If we had more time, we might explore dropping `sqft_living15` instead.

6. Residual Analysis

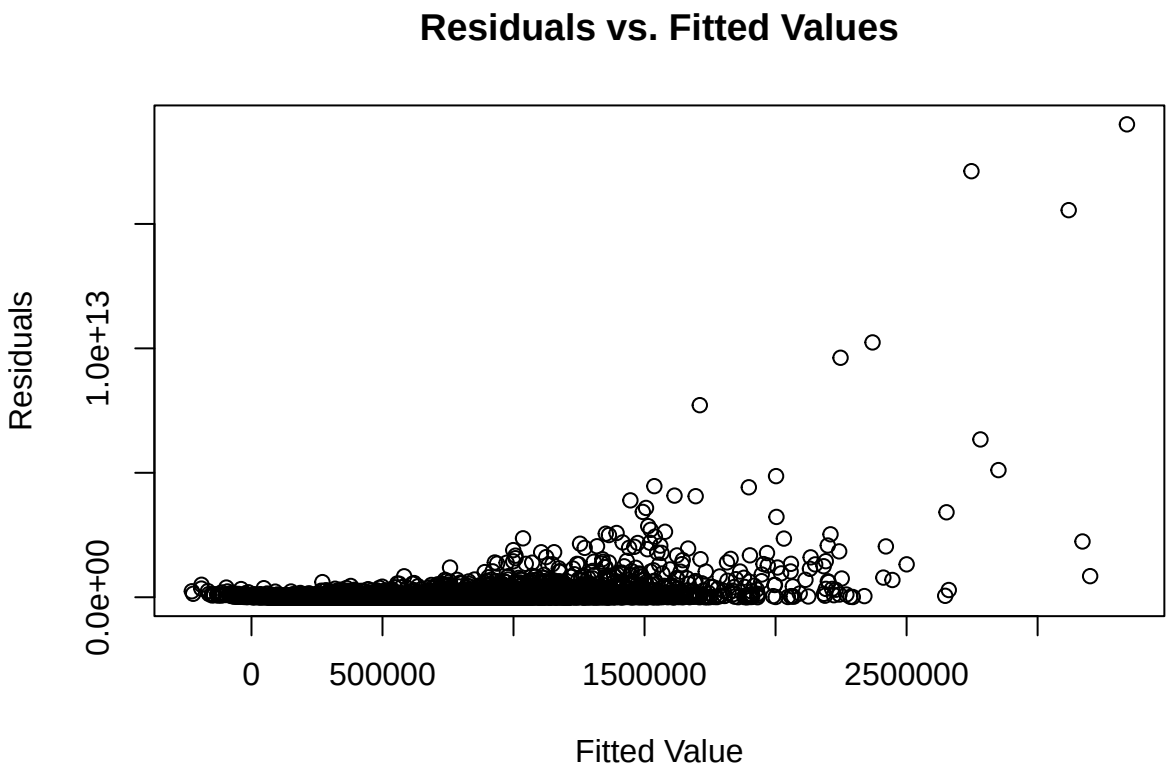
```
# Plot residuals vs fitted
plot(fitted(model.noncol), resid(model.noncol), xlab = "Fitted Value",
    ylab = "Residuals", main = "Residuals vs. Fitted Values")
abline(0, 0, col = "red")
```



From the plot of the residuals vs fitted, we can see that the data is very roughly centered around the y=0 line, though it possibly exhibits a weak quadratic trend. We may want to add quadratic or interaction terms to improve our residual fit. We can certainly see that the data is not normally distributed with constant variance. Our residuals show strong evidence of heteroskedasticity.

For good measure, we also plot the squared residuals against fitted values:

```
# Plot squared residuals vs fitted
plot(fitted(model.noncol), resid(model.noncol)^2, xlab = "Fitted Value",
    ylab = "Residuals", main = "Residuals vs. Fitted Values")
```



The squared residuals confirm heteroskedasticity, which we will explore later.

7. Model Specification

```
resettest(model.noncol, power=2:3, type="regressor", data=kc_house)
```

```
##
##  RESET test
##
## data:  model.noncol
## RESET = 221.35, df1 = 36, df2 = 21541, p-value < 2.2e-16
```

The RESET test shows very strong evidence that we are missing some polynomial terms. It would be difficult for us to consider every possible polynomial and interaction term with 13 features to choose from, so we will opt for a few polynomial terms that have an explainable economic rationale. If we had more time, we could explore the model space more fully to find a better-specified model.

8. Heteroskedasticity

8.1 Breusch-Pagan Test

```
bptest(model.noncol)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model.noncol
## BP = 2747.1, df = 13, p-value < 2.2e-16
```

The Breusch-Pagan test confirms what our residual plot suggested: there is no doubt that our data is heteroskedastic.

8.2 Feasible Generalized Least Squares

We will use the feasible generalized least squares method to address the heteroskedasticity in our data. Based on the previous residual plot, we determine that an exponential model of the variance is appropriate.

```
ehatsq <- resid(model.noncol)^2
# Only select columns with positive features so log is finite and defined
sighatsq.ols <- lm(log(ehatsq) ~ log(bedrooms) + log(bathrooms) + log(sqft_living) +
                     log(sqft_living15) + log(sqft_lot15), data=kc_house)
vari <- exp(fitted(sighatsq.ols))
model.fgls <- lm(price ~ . -sqft_lot -floors -spr_sum -sqft_above -yr_renovated,
                 weights=1/vari, data=kc_house)
summary(model.fgls)
```

```
##
## Call:
## lm(formula = price ~ . - sqft_lot - floors - spr_sum - sqft_above -
##     yr_renovated, data = kc_house, weights = 1/vari)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -10.2251  -1.3000  -0.3011   0.8698  19.2727
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -7.884e+04  1.367e+04  -5.767 8.20e-09 ***
## bedrooms      -2.604e+04  1.448e+03 -17.981  < 2e-16 ***
## bathrooms      3.150e+04  2.232e+03  14.113  < 2e-16 ***
## sqft_living    1.267e+02  2.950e+00  42.940  < 2e-16 ***
## waterfront     3.655e+05  1.660e+04  22.018  < 2e-16 ***
## view           4.437e+04  1.884e+03  23.552  < 2e-16 ***
## condition      2.117e+04  1.509e+03  14.032  < 2e-16 ***
## grade          8.400e+04  1.558e+03  53.920  < 2e-16 ***
## yr_built      -1.980e+03  4.518e+01 -43.835  < 2e-16 ***
## lat            5.020e+05  7.237e+03  69.360  < 2e-16 ***
```

```
## long           4.368e+04  7.980e+03   5.474 4.45e-08 ***
## sqft_living15  4.930e+01  2.736e+00  18.020  < 2e-16 ***
## sqft_lot15    -1.612e-01  5.737e-02  -2.809  0.00497 **
## yr_sold        2.622e+04  2.012e+03  13.031  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.074 on 21577 degrees of freedom
## Multiple R-squared:  0.6706, Adjusted R-squared:  0.6704
## F-statistic:  3379 on 13 and 21577 DF,  p-value: < 2.2e-16
```

After accounting for the heteroskedasticity, almost all of the standard errors have improved by some amount, except for `sqft_lot15`. The adjusted R^2 has gone down slightly. Almost all of the coefficients have moved towards 0, which may mean that the model is more highly weighting the majority of less extreme values and placing less emphasis on the examples with extreme variance, as we expect from FGLS.

9. Polynomial Regression

To address our concerns from the RESET test, we include two higher-order terms with potential economic significance:

- `sqft_living` squared: It is possible that there are diminishing returns to living space square footage
- `waterfront` * `sqft_living`: Houses located near the waterfront may have a higher price per square foot (not just a higher intercept)

```
model.poly <- lm(price ~ . +I(sqft_living^2) +I(waterfront*sqft_living) -sqft_lot
                 -floors -spr_sum -sqft_above -yr_renovated, weights=1/vari,
                 data=kc_house)
summary(model.poly)
```

```
##
## Call:
## lm(formula = price ~ . + I(sqft_living^2) + I(waterfront * sqft_living) -
##     sqft_lot - floors - spr_sum - sqft_above - yr_renovated,
##     data = kc_house, weights = 1/vari)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -15.3328  -1.2362  -0.2318   0.9181  16.7998
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  3.496e+04  1.347e+04   2.596  0.00945 **
## bedrooms                    -1.203e+04  1.429e+03  -8.422  < 2e-16 ***
## bathrooms                    4.103e+04  2.149e+03  19.095  < 2e-16 ***
## sqft_living                 -5.793e+01  5.636e+00 -10.279  < 2e-16 ***
## waterfront                  -1.722e+05  3.000e+04  -5.742 9.47e-09 ***
## view                         4.128e+04  1.803e+03  22.893  < 2e-16 ***
## condition                    2.409e+04  1.444e+03  16.684  < 2e-16 ***
## grade                        8.571e+04  1.492e+03  57.463  < 2e-16 ***
## yr_built                    -1.887e+03  4.324e+01 -43.629  < 2e-16 ***
```

```
## lat                           4.978e+05  6.920e+03  71.933  < 2e-16 ***
## long                          3.929e+04  7.636e+03   5.145 2.70e-07 ***
## sqft_living15                 5.451e+01  2.618e+00  20.821  < 2e-16 ***
## sqft_lot15                   -1.629e-01  5.488e-02  -2.969  0.00300 **
## yr_sold                       2.591e+04  1.924e+03  13.467  < 2e-16 ***
## I(sqft_living^2)              3.738e-02  1.017e-03  36.770  < 2e-16 ***
## I(waterfront * sqft_living)   2.413e+02  1.169e+01  20.638  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.983 on 21575 degrees of freedom
## Multiple R-squared:  0.699,  Adjusted R-squared:  0.6988
## F-statistic:  3340 on 15 and 21575 DF,  p-value: < 2.2e-16
```

It appears that both high-order terms are significant, though the coefficient for `sqft_living` squared is very small. It appears that our intuition about waterfront square footage was correct. This model has a slightly better adjusted R^2 than the previous FGLS model without high-order terms. It is possible that we could continue to improve the model with additional high-order terms.

Economic Interpretation

With the addition of high order terms, some of our coefficients have changed their economic interpretations:

- The intercept is now (weakly) positive
- `sqft_living` is now negative, but `sqft_living` squared is slightly positive. This would indicate that houses with more square footage have an increasing marginal value. However, the mildly collinear feature `sqft_living15` that we elected to keep still has a positive coefficient, so we may have ambiguity caused by collinearity.
- `waterfront` now has a large negative value, but the interaction term with `sqft_living` is very positive. This indicates that our assumption was correct: Houses located near the waterfront have a higher price per square foot.

The addition of higher-order terms has made the economic interpretation of our model more difficult in some respect (e.g. a negative value for `sqft_living`), but it provides evidence for other economic hypotheses we had and significantly increases the predictive power of our model. We now attempt to quantify that improvement by comparing against our previous models.

9.1 Model Comparison

```
models = list(
  "baseline"=model.base,
  "baseline (no outliers)"=model.rebase,
  "feature selection"=model.featsel,
  "collinearity removed"=model.noncol,
  "fgls"=model.fgls,
  "polynomial terms"=model.poly)

performance = data.frame(AIC=sapply(models, AIC), BIC=sapply(models, BIC))
stargazer(performance, summary=FALSE, type="text")
```

```
##
## ============================================
##                             AIC         BIC
## --------------------------------------------
## baseline                589,446.300 589,605.900
## baseline (no outliers)  588,778.700 588,938.300
## feature selection       588,790.600 588,926.300
## collinearity removed    588,886.300 589,006.000
## fgls                    576,696.900 576,816.600
## polynomial terms        574,757.800 574,893.400
## --------------------------------------------
```

We can clearly see that the AIC and BIC are decreasing with every model improvement we made. While the starkest jump is from switching to FGLS, our polynomial terms improved the model by just under 2,000 units on both AIC and BIC. This is a good improvement, and indicates that we have made judicious choices to refine our model.

10. Performance

10.1 Train Test Split

```
house_features <- subset(kc_house, select=-c(sqft_lot, floors, spr_sum,
                                             sqft_above, yr_renovated))
# Use a 2/3-1/3 train-test split
split <- sample(1:nrow(house_features), 0.67*nrow(house_features))
train <- house_features[split, ]
train_weights <- vari[split]
test <- house_features[-split, ]
test_weights <- vari[split]
dim(train)
```

```
## [1] 14465    14
```

```
dim(test)
```

```
## [1] 7126   14
```

```
reg.train <- lm(price ~ . +I(sqft_living^2) +I(waterfront*sqft_living),
                weights=1/train_weights, data=train)
summary(reg.train)
```

```
##
## Call:
## lm(formula = price ~ . + I(sqft_living^2) + I(waterfront * sqft_living),
##     data = train, weights = 1/train_weights)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5877  -1.2321  -0.2178   0.9274  16.8145
```

```
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   3.471e+04  1.636e+04   2.122   0.0339 *
## bedrooms                     -1.086e+04  1.719e+03  -6.318 2.72e-10 ***
## bathrooms                     4.190e+04  2.600e+03  16.114  < 2e-16 ***
## sqft_living                  -6.474e+01  6.736e+00  -9.610  < 2e-16 ***
## waterfront                   -2.129e+05  3.646e+04  -5.838 5.39e-09 ***
## view                          4.334e+04  2.196e+03  19.734  < 2e-16 ***
## condition                     2.352e+04  1.750e+03  13.441  < 2e-16 ***
## grade                         8.617e+04  1.815e+03  47.466  < 2e-16 ***
## yr_built                     -1.892e+03  5.197e+01 -36.415  < 2e-16 ***
## lat                           4.995e+05  8.335e+03  59.933  < 2e-16 ***
## long                          4.165e+04  9.301e+03   4.479 7.57e-06 ***
## sqft_living15                 5.424e+01  3.162e+00  17.154  < 2e-16 ***
## sqft_lot15                   -1.442e-01  6.407e-02  -2.252   0.0244 *
## yr_sold                       2.570e+04  2.329e+03  11.036  < 2e-16 ***
## I(sqft_living^2)              3.817e-02  1.206e-03  31.652  < 2e-16 ***
## I(waterfront * sqft_living)   2.647e+02  1.412e+01  18.742  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.966 on 14449 degrees of freedom
## Multiple R-squared:  0.7048, Adjusted R-squared:  0.7045
## F-statistic:  2300 on 15 and 14449 DF,  p-value: < 2.2e-16
```

```
sqrt(mean((train$price - predict(reg.train, train)) ^ 2))
```

```
## [1] 191258.9
```

```
sqrt(mean((test$price - predict(reg.train, test)) ^ 2))
```

```
## [1] 187796.7
```

The MSE values for our train and test sets are relatively large, representing about 35% of the average home price. However, when we consider the large range of values that `price` takes on (over $7M), this RMSE seems more reasonable. The RMSE value for our test set is comparable to the training RMSE, which means that our model is not overfitting to the training data and is capable of generalization on out-of-sample observations.

10.2 5-Fold Cross-Validation

```
fit <- lm(price ~ . +I(sqft_living^2) +I(waterfront*sqft_living),
                weights=1/vari, data=kc_house, x=TRUE, y=TRUE)
cv.lm(fit, k=5)
```

```
## Mean absolute error        :  117640.4
## Sample standard deviation  :  1974.809
##
## Mean squared error         :  36275162301
```

```
## Sample standard deviation  :  4062293027
##
## Root mean squared error    :  190216.6
## Sample standard deviation  :  10769.89
```

Using 5-Fold Cross Validation, we can evaluate our model's performance even better. The RMSE is very similar to the RMSE we got in our single train-test split, and with a relatively stable standard deviation. However, we can see that the MAE is significantly lower than the RMSE. Since we know that RMSE is much more sensitive to values far from the mean, this indicates that our model performs better on most of the values in the dataset compared to the extreme tails. This confirms what we originally commented in our exploratory data analysis, that the extremely long tail of `price` is more difficult to model. The MAE is about 25% of the mean home value, which, while it is a large prediction error, is still in the "ballpark" of home values and indicates that our model performs at least adequately, given the dataset and specification.

11. Conclusions

Our overall conclusions are that multiple linear regression performs relatively mediocre for house price prediction. We found a number of strong predictors with straightforward economic interpretations, including the size of the home, its location, quality of construction, and age. We were able to significantly reduce the number of predictors required from 18 to 13 through prudent variable selection without losing accuracy. We were able to improve our model's predictive accuracy by including a select few higher-order polynomial and interaction terms, but this resulted in a model that was more difficult to assign an economic interpretation to. Our final model performed adequately, but as we discovered in our analysis, the underlying data was highly heteroskedastic which led to large prediction errors for our linear model. Even with FGLS, the mean squared error of our predictions was still relatively high. From examining 5-fold cross validation results, we found that our model performed much better on the majority of the dataset, and worse on the extreme tail of very expensive homes.

11.1 Future Work

Our work leaves many open questions that could be explored in the future. We weren't able to rigorously explore all the polynomial and interaction terms that could be included in our model. The model space is very large, and it would take more time than we have available for this assignment.

Additional feature engineering could also yield dividends. For example, we noticed that some of the predictor variables appeared to be log-normally distributed, though we declined to log transform them.