8.2Exercises.R

Rahul Rajeev

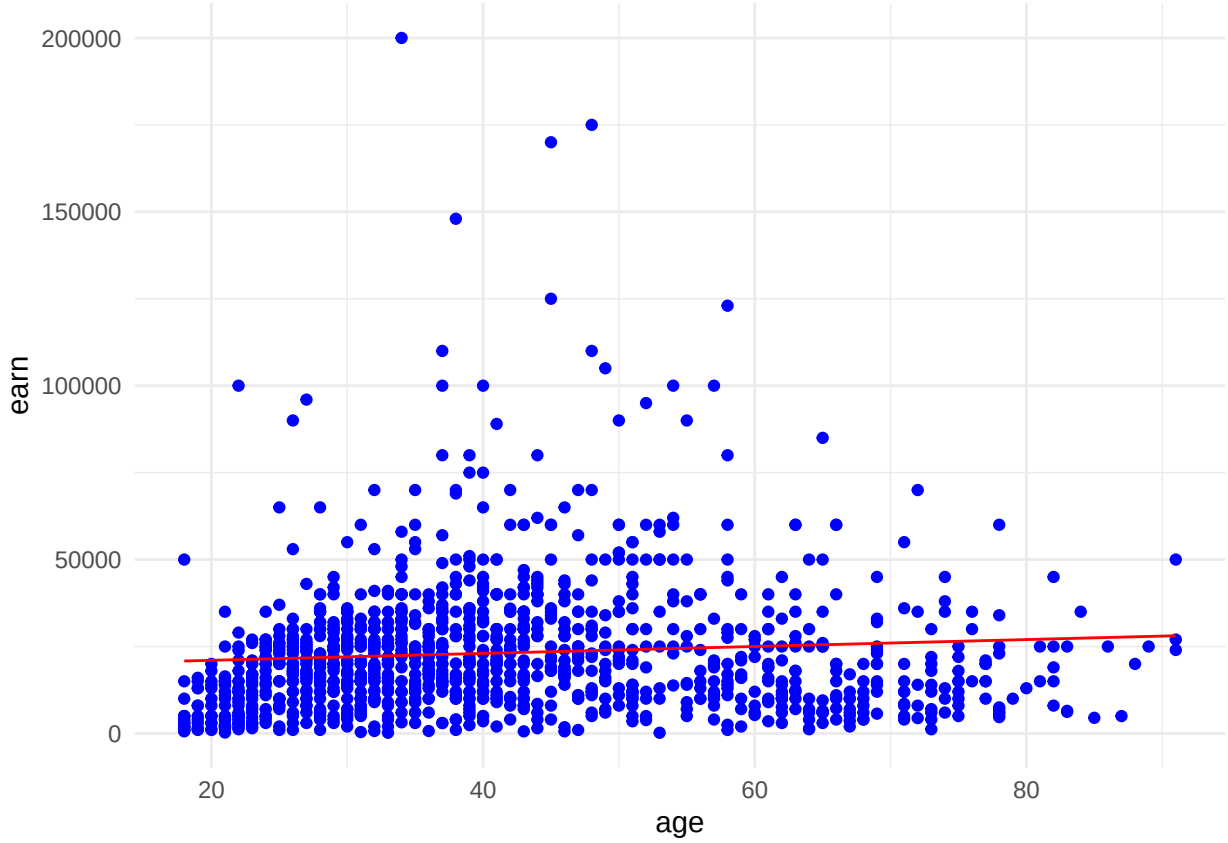
2023-02-02

```
# Assignment: 8.2 Exercises - Assignment 6
# Name: Rajeev, Rahul
# Date: 2023-01-30

## Set the working directory to the root of your DSC 520 directory
library(ggplot2)
theme_set(theme_minimal())

## Set the working directory to the root of your DSC 520 directory
setwd("C:/Users/rahul/Documents/Bellevue/DSC 520")

## Load the `data/r4ds/heights.csv` to
heights_df <- read.csv("data/r4ds/heights.csv")

## Fit a linear model using the `age` variable as the predictor
# and `earn` as the outcome
age_lm <-  lm(heights_df$earn~heights_df$age)

## View the summary of your model using `summary()`
summary(age_lm)
```

```
##
## Call:
## lm(formula = heights_df$earn ~ heights_df$age)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -25098 -12622  -3667   6883 177579
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    19041.53    1571.26  12.119  < 2e-16 ***
## heights_df$age    99.41      35.46   2.804  0.00514 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19420 on 1190 degrees of freedom
## Multiple R-squared:  0.006561,   Adjusted R-squared:  0.005727
## F-statistic:  7.86 on 1 and 1190 DF,  p-value: 0.005137
```

```
## Creating predictions using `predict()`
age_predict_df <- data.frame(earn =
                               predict(age_lm, heights_df), age=heights_df$age)
```

```r
## Plot the predictions against the original data
ggplot(data =  heights_df, aes(y = earn, x = age)) +
  geom_point(color='blue') +
  geom_line(color='red',data = age_predict_df, aes(y= earn, x=age))
```




```r
mean_earn <- mean(heights_df$earn)
## Corrected Sum of Squares Total
sst <- sum((mean_earn - heights_df$earn)^2)
## Corrected Sum of Squares for Model
ssm <- sum((mean_earn - age_predict_df$earn)^2)
## Residuals
residuals <- heights_df$earn - age_predict_df$earn
## Sum of Squares for Error
sse <- sum(residuals^2)
## R Squared R^2 = SSM\SST
r_squared <- ssm/sst

## Number of observations
n <- length(heights_df$age)
## Number of regression parameters
p <- 2
## Corrected Degrees of Freedom for Model (p-1)
dfm <- p-1
## Degrees of Freedom for Error (n-p)
dfe <- n-p
```

```r
## Corrected Degrees of Freedom Total:   DFT = n - 1
dft <- n-1

## Mean of Squares for Model:   MSM = SSM / DFM
msm <- ssm/dfm
## Mean of Squares for Error:   MSE = SSE / DFE
mse <- sse/dfe
## Mean of Squares Total:   MST = SST / DFT
mst <- sst/dft
## F Statistic F = MSM/MSE
f_score <- msm/mse

## Adjusted R Squared R2 = 1 - (1 - R2)(n - 1) / (n - p)
adjusted_r_squared <- 1 - (1 - r_squared)*(n - 1)/(n - p)

## Calculate the p-value from the F distribution
p_value <- pf(f_score, dfm, dft, lower.tail=F)
```

```r
# Assignment: 8.2 Exercises - Assignment 7
# Name: Rajeev, Rahul
# Date: 2023-30-01

# Fit a linear model
earn_lm <-  lm(earn ~ height + sex + ed + age + race, data=heights_df)

# View the summary of your model
summary(earn_lm)
```

```
##
## Call:
## lm(formula = earn ~ height + sex + ed + age + race, data = heights_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -39423  -9827  -2208   6157 158723
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -41478.4    12409.4  -3.342 0.000856 ***
## height          202.5      185.6   1.091 0.275420
## sexmale       10325.6     1424.5   7.249 7.57e-13 ***
## ed             2768.4      209.9  13.190  < 2e-16 ***
## age             178.3       32.2   5.537 3.78e-08 ***
## racehispanic  -1414.3     2685.2  -0.527 0.598507
## raceother       371.0     3837.0   0.097 0.922983
## racewhite      2432.5     1723.9   1.411 0.158489
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17250 on 1184 degrees of freedom
## Multiple R-squared:  0.2199, Adjusted R-squared:  0.2153
## F-statistic: 47.68 on 7 and 1184 DF,  p-value: < 2.2e-16
```

```
predicted_df <- data.frame(earn = predict(earn_lm, heights_df),
                           ed=heights_df$ed, race=heights_df$race,
                           height=heights_df$height, age=heights_df$age,
                           sex=heights_df$sex)

## Compute deviation (i.e. residuals)
mean_earn <- mean(heights_df$earn)
## Corrected Sum of Squares Total
sst <- sum((mean_earn - heights_df$earn)^2)
## Corrected Sum of Squares for Model
ssm <- sum((mean_earn - predicted_df$earn)^2)
## Residuals
residuals <- heights_df$earn - predicted_df$earn
## Sum of Squares for Error
sse <- sum(residuals^2)
## R Squared
r_squared <- ssm/sst
r_squared
```

```
## [1] 0.2198953
```

```
## Number of observations
n <- length(heights_df$earn)
## Number of regression paramaters
p <- 8
## Corrected Degrees of Freedom for Model
dfm <- p - 1
## Degrees of Freedom for Error
dfe <- n - p
## Corrected Degrees of Freedom Total:   DFT = n - 1
dft <- n - 1

## Mean of Squares for Model:   MSM = SSM / DFM
msm <- ssm/dfm
## Mean of Squares for Error:   MSE = SSE / DFE
mse <- sse/dfe
## Mean of Squares Total:   MST = SST / DFT
mst <- sst/dft
## F Statistic
f_score <- msm/mse

## Adjusted R Squared R2 = 1 - (1 - R2)(n - 1) / (n - p)
adjusted_r_squared <- 1 - (1 - r_squared)*(n - 1)/(n - p)
```

```
# Assignment: 8.2 Exercise - Housing Dataset
# Name: Rajeev, Rahul
# Date: 2023-01-30

# importing libraries
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(readxl)
library(pastecs)
```

```
##
## Attaching package: 'pastecs'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
library(scales)
library(purrr)
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:scales':
##
##     discard

## The following object is masked from 'package:plyr':
##
##     compact
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following object is masked from 'package:pastecs':
##
##     extract
```

```
# loading dataset
housing_xl <- read_excel("data/week-7-housing.xlsx")
housing_df <- data.frame(housing_xl)

# just checking whether the data frame loaded properly
head(housing_df)
```

```
##    Sale.Date Sale.Price sale_reason sale_instrument sale_warning sitetype
## 1 2006-01-03     698000           1               3         <NA>       R1
## 2 2006-01-03     649990           1               3         <NA>       R1
## 3 2006-01-03     572500           1               3         <NA>       R1
## 4 2006-01-03     420000           1               3         <NA>       R1
## 5 2006-01-03     369900           1               3           15       R1
## 6 2006-01-03     184667           1              15        18 51       R1
```

```
##            addr_full  zip5 ctyname postalctyn       lon      lat building_grade
## 1  17021 NE 113TH CT 98052 REDMOND    REDMOND -122.1124 47.70139              9
## 2  11927 178TH PL NE 98052 REDMOND    REDMOND -122.1022 47.70731              9
## 3 13315 174TH AVE NE 98052    <NA>    REDMOND -122.1085 47.71986              8
## 4  3303 178TH AVE NE 98052 REDMOND    REDMOND -122.1037 47.63914              8
## 5  16126 NE 108TH CT 98052 REDMOND    REDMOND -122.1242 47.69748              7
## 6   8101 229TH DR NE 98053    <NA>    REDMOND -122.0341 47.67545              7
##   square_feet_total_living bedrooms bath_full_count bath_half_count
## 1                     2810        4               2               1
## 2                     2880        4               2               0
## 3                     2770        4               1               1
## 4                     1620        3               1               0
## 5                     1440        3               1               0
## 6                     4160        4               2               1
##   bath_3qtr_count year_built year_renovated current_zoning sq_ft_lot prop_type
## 1               0       2003              0             R4      6635         R
## 2               1       2006              0             R4      5570         R
## 3               1       1987              0             R6      8444         R
## 4               1       1968              0             R4      9600         R
## 5               1       1980              0             R6      7526         R
## 6               1       2005              0          URPSO      7280         R
##   present_use
## 1           2
## 2           2
## 3           2
## 4           2
## 5           2
## 6           2
```

```
## i) Explain any transformations or modifications you made to the dataset

# filtering data
filtered_housing_df <- housing_df %>% filter(Sale.Price > 100000)
length(housing_df$Sale.Price)
```

```
## [1] 12865
```

```
length(filtered_housing_df$Sale.Price)
```

```
## [1] 12786
```

```
# filtered out house prices with less than 100,000 to avoid outliers that don't
# make any sense

# selecting columns that would be useful for analysis
selected_housing_df <- filtered_housing_df %>% select(Sale.Price,
                                                      square_feet_total_living,
                                                      bedrooms,
                                                      sq_ft_lot,
                                                      year_built,
                                                      building_grade)
head(selected_housing_df)
```

```
##   Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 1     698000                     2810        4      6635       2003
## 2     649990                     2880        4      5570       2006
```

```
## 3     572500                    2770        4      8444       1987
## 4     420000                    1620        3      9600       1968
## 5     369900                    1440        3      7526       1980
## 6     184667                    4160        4      7280       2005
##   building_grade
## 1              9
## 2              9
## 3              8
## 4              8
## 5              7
## 6              7
```

```
## ii) creating variables and predictors

# creating a variable for linear model of sale price vs square foot lot
saleprice_lm <-  lm(Sale.Price ~
                      sq_ft_lot, data=selected_housing_df)

# creating a variable for linear model of sale price vs additional variables
multi_lm <- lm(Sale.Price ~ sq_ft_lot + square_feet_total_living +
                 bedrooms + year_built + building_grade, data=selected_housing_df)

# I chose square feet total living, bedrooms and year_built as additional
# parameters because I feel like they are all factors that influence
# a house price. Having bigger, newer homes I expect should be the costliest,
# but I also think that some ancient houses may be also costly depending on
# renovation. Since there isn't a great variable for renovation in the dataset
# I can't really use it.

## iii) execute summary to compare model results

# summary of single linear regression
summary(saleprice_lm)
```

```
##
## Call:
## lm(formula = Sale.Price ~ sq_ft_lot, data = selected_housing_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1434837  -194636   -64399    89447  3736154
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.394e+05  3.807e+03  167.98   <2e-16 ***
## sq_ft_lot   1.166e+00  6.721e-02   17.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 398000 on 12784 degrees of freedom
## Multiple R-squared:  0.02298,    Adjusted R-squared:  0.02291
## F-statistic: 300.7 on 1 and 12784 DF,  p-value: < 2.2e-16
```

```
# R-squared is 0.00298, and the adjusted r-squared is 0.02291
```

```r
# summary of multiregression
summary(multi_lm)
```

```
##
## Call:
## lm(formula = Sale.Price ~ sq_ft_lot + square_feet_total_living +
##     bedrooms + year_built + building_grade, data = selected_housing_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1409171  -119480   -43556    40309  3701965
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -5.294e+06  4.053e+05 -13.065  < 2e-16 ***
## sq_ft_lot                 5.581e-01  6.388e-02   8.737  < 2e-16 ***
## square_feet_total_living  1.450e+02  5.790e+00  25.040  < 2e-16 ***
## bedrooms                 -3.982e+03  4.557e+03  -0.874    0.382
## year_built                2.685e+03  2.054e+02  13.070  < 2e-16 ***
## building_grade            2.930e+04  4.425e+03   6.621 3.71e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 353000 on 12780 degrees of freedom
## Multiple R-squared:  0.2318, Adjusted R-squared:  0.2315
## F-statistic: 771.3 on 5 and 12780 DF,  p-value: < 2.2e-16
```

```r
# the r-squared statistic has increased to 0.2292, and the adjusted r-squared
# is 0.2289.

# significance of the difference between models in r-squared and adjusted
# r-squared:

# Since the model we are using is trying to predict the variance in sale price
# based on a number of factors, we would need a larger r-squared to demonstrate
# that the model shows this variance.
# The simple linear model has a very low r-squared and adjusted r-squared values
# compared to the multiregression model, leading me to believe that the
# multi-regression model that includes the extra parameters fits the data
# better.

## iv) calculating the standard beta coefficients for the multiregression model

# standardized coefficients
library(lm.beta)
lm.beta(multi_lm)
```

```
##
## Call:
## lm(formula = Sale.Price ~ sq_ft_lot + square_feet_total_living +
##     bedrooms + year_built + building_grade, data = selected_housing_df)
##
## Standardized Coefficients::
##              (Intercept)                sq_ft_lot square_feet_total_living
##                       NA              0.072598555              0.355285707
```

```
##                 bedrooms                year_built           building_grade
##             -0.008662599                0.114388648               0.079311317
```

```
# scaling the original model to check whether the function works
standard_multi_lm <- lm(scale(Sale.Price) ~ scale(sq_ft_lot) +
                          scale(square_feet_total_living) +
                          scale(bedrooms) + scale(year_built) +
                          scale(building_grade), data=selected_housing_df)
summary(standard_multi_lm)
```

```
##
## Call:
## lm(formula = scale(Sale.Price) ~ scale(sq_ft_lot) + scale(square_feet_total_living) +
##     scale(bedrooms) + scale(year_built) + scale(building_grade),
##     data = selected_housing_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4998 -0.2967 -0.1082  0.1001  9.1941
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -1.152e-15  7.753e-03   0.000    1.000
## scale(sq_ft_lot)                7.260e-02  8.309e-03   8.737  < 2e-16 ***
## scale(square_feet_total_living)  3.553e-01  1.419e-02  25.040  < 2e-16 ***
## scale(bedrooms)                -8.663e-03  9.913e-03  -0.874    0.382
## scale(year_built)               1.144e-01  8.752e-03  13.070  < 2e-16 ***
## scale(building_grade)           7.931e-02  1.198e-02   6.621 3.71e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8766 on 12780 degrees of freedom
## Multiple R-squared:  0.2318, Adjusted R-squared:  0.2315
## F-statistic: 771.3 on 5 and 12780 DF,  p-value: < 2.2e-16
```

```
# looks like we are good to go!

# the beta coefficients for each variable in the multi-regression model
# are shown above. From a glance, all variables
# have a positive coefficient except for bedrooms, which has a negative. The
# positive coefficients show that there is a positive relationship between
# sq_ft_lot, square_feet_total_living, year_built, and building_grade on
# the sale.price of the house. The negative relationship between bedrooms and
# sale price is interesting considering I would've expected that more bedrooms
# would have resulted in a higher house price


## v) confidence intervals for each of the parameters in the model

confint(standard_multi_lm, level = 0.95)
```

```
##                                     2.5 %     97.5 %
## (Intercept)                    -0.01519643 0.01519643
## scale(sq_ft_lot)                0.05631186 0.08888525
## scale(square_feet_total_living)  0.32747421 0.38309720
## scale(bedrooms)                -0.02809265 0.01076745
```

```
## scale(year_built)                0.09723380 0.13154350
## scale(building_grade)            0.05583067 0.10279196
```

```
# the confidence intervals are pretty narrow except for bedrooms which suggests
# that 95% of all samples that can be drawn, the beta coefficients of
# the other variables will cover the true value in relationship to the entire
# regression model.

## vi) assess the improvement of the new model using an analysis of variance
anova(saleprice_lm, multi_lm)
```

```
## Analysis of Variance Table
##
## Model 1: Sale.Price ~ sq_ft_lot
## Model 2: Sale.Price ~ sq_ft_lot + square_feet_total_living + bedrooms +
##     year_built + building_grade
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
## 1  12784 2.0251e+15
## 2  12780 1.5923e+15  4 4.3284e+14 868.52 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Null hypothesis: there is no difference in the models, adding more variables
# does not improve the relationship with Sale Price.
# Alternative hypothesis: The coefficients of both models are not the same

# Setting a significance level of 0.05. The F-statistic of the multi-linear
# model is 868.52 on 5 and with 12780 DF, while the F-statistic of the linear
# model is 300 on 1 and 12784 DF

# if the null hypothesis were true, the F-statistic of each model would be
# closer to 1 which they aren't. Although both have very small p-values below
# the significance level, the multi-regression model has a higher f-statistic
# which means that there is larger variation between the models than within
# the models. Which leads to my conclusion that the multi-regression model
# fits the data better.

## vii) outliers and influential cases
outlier_points <- hatvalues(multi_lm) > 3 * mean(hatvalues(multi_lm))
outliers <- selected_housing_df[outlier_points,]
head(outliers)
```

```
##     Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 6       184667                     4160        4      7280       2005
## 14      165000                     1850        3    278891       2011
## 65      446400                     1770        3    220654       1984
## 72     1900000                     6610        4     37017       1990
## 92      732500                     5710        5     10200       1977
## 108    1520000                     4640        5     19173       1952
##     building_grade
## 6                7
## 14               9
## 65               7
## 72              11
## 92               9
## 108              9
```

```
length(outliers$Sale.Price)
```

```
## [1] 447
```

```
# there are 447 outliers

cooks_crit = 0.5
influential_points <- cooks.distance(multi_lm)
influential_cases <- selected_housing_df[which(abs(influential_points) > cooks_crit),]
influential_cases
```

```
##      Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 4608    4400000                     2410        3   1327090       1935
##      building_grade
## 4608              6
```

```
# there is one influential case

## viii) calculate standardized residuals
std_res <- rstandard(multi_lm)

# storing results
selected_res_df <- cbind(selected_housing_df, std_res)
head(selected_res_df)
```

```
##   Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 1     698000                     2810        4      6635       2003
## 2     649990                     2880        4      5570       2006
## 3     572500                     2770        4      8444       1987
## 4     420000                     1620        3      9600       1968
## 5     369900                     1440        3      7526       1980
## 6     184667                     4160        4      7280       2005
##   building_grade     std_res
## 1              9 -0.12559062
## 2              9 -0.31151074
## 3              8 -0.26286734
## 4              8 -0.09114736
## 5              7 -0.16414591
## 6              7 -1.98603616
```

```
# ordering to look at magnitude of residuals
head(selected_res_df[order(-std_res),])
```

```
##       Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 6377     4380542                     2450        4      4749       2010
## 6385     4380542                     2440        3      4244       2011
## 6384     4380542                     2550        4      4368       2010
## 6378     4380542                     2750        4      5816       2012
## 11915    4311000                     1670        3    425145       1964
## 6383     4380542                     2810        4     13289       2012
##       building_grade  std_res
## 6377               8 10.48928
## 6385               8 10.47503
## 6384               8 10.44874
## 6378               8 10.34918
## 11915              8 10.31354
```

```
## 6383               8 10.31272
```

```
# picking points that have large residuals > or < than +2, -2
large_res_df <- subset(selected_res_df, selected_res_df$std_res > 2 |
                                  selected_res_df$std_res < -2)

# checking lengths
length(selected_res_df$Sale.Price)
```

```
## [1] 12786
```

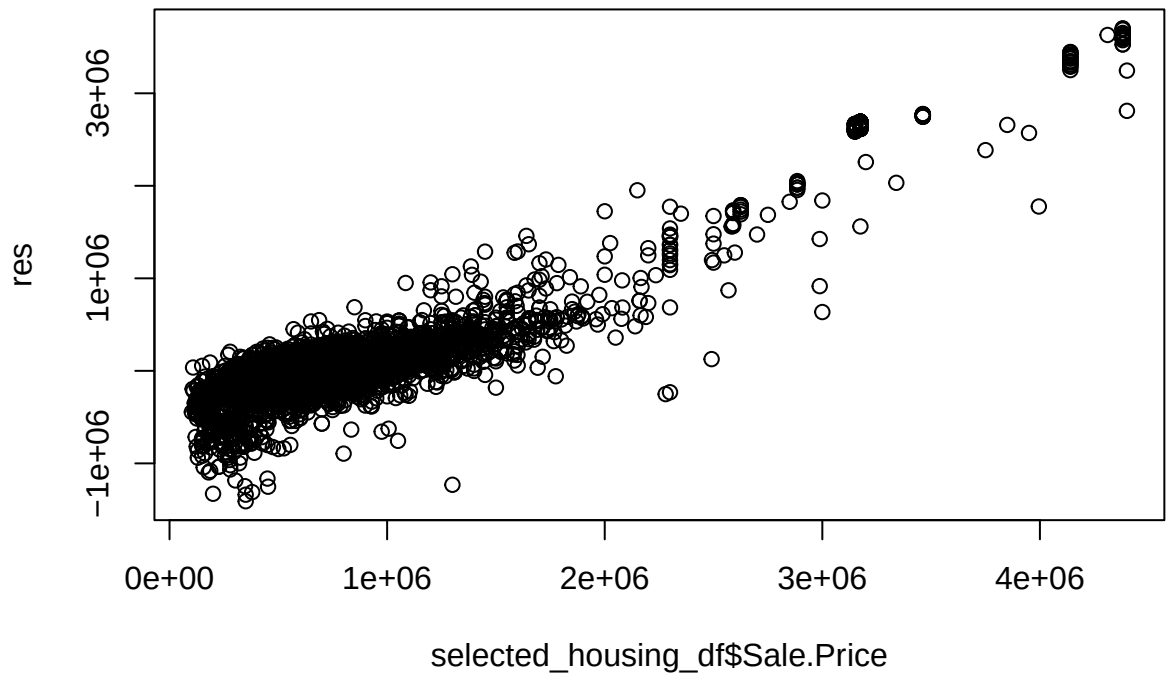
```
length(large_res_df$Sale.Price)
```

```
## [1] 321
```

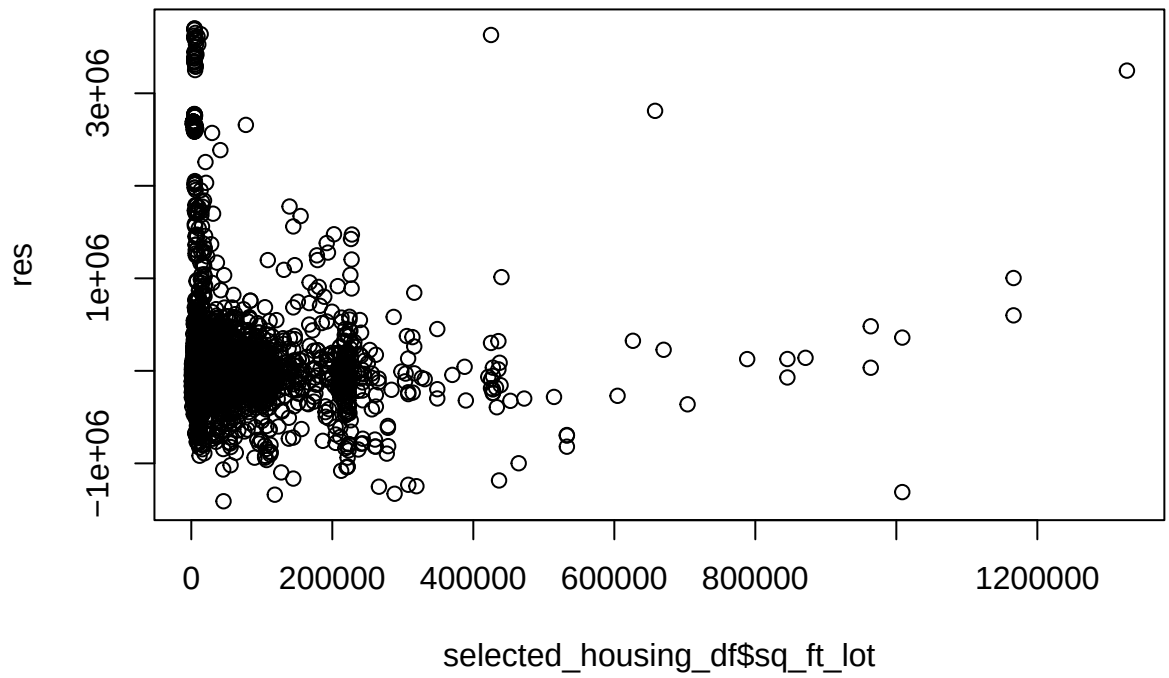
```
# there are 321 points with large residuals

## ix) show the sum of large residuals
sum(large_res_df$std_res)
```

```
## [1] 1237.564
```

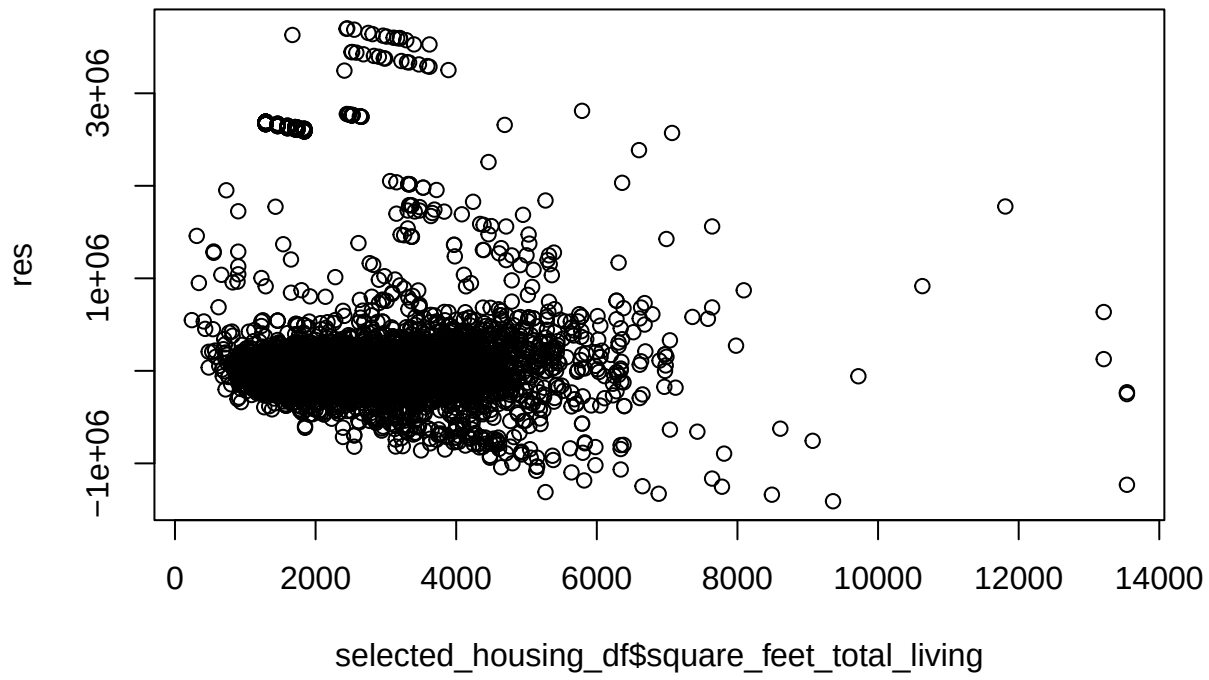
```
# the sum of the large standardized residuals is 1237.564

## x) what specific variables have large residuals
res = resid(multi_lm)

# I plotted the residuals of the model vs each variable and found the variables
# with the largest residuals are sale price, square feet lot,
# and square foot living

plot(selected_housing_df$Sale.Price, res)
```



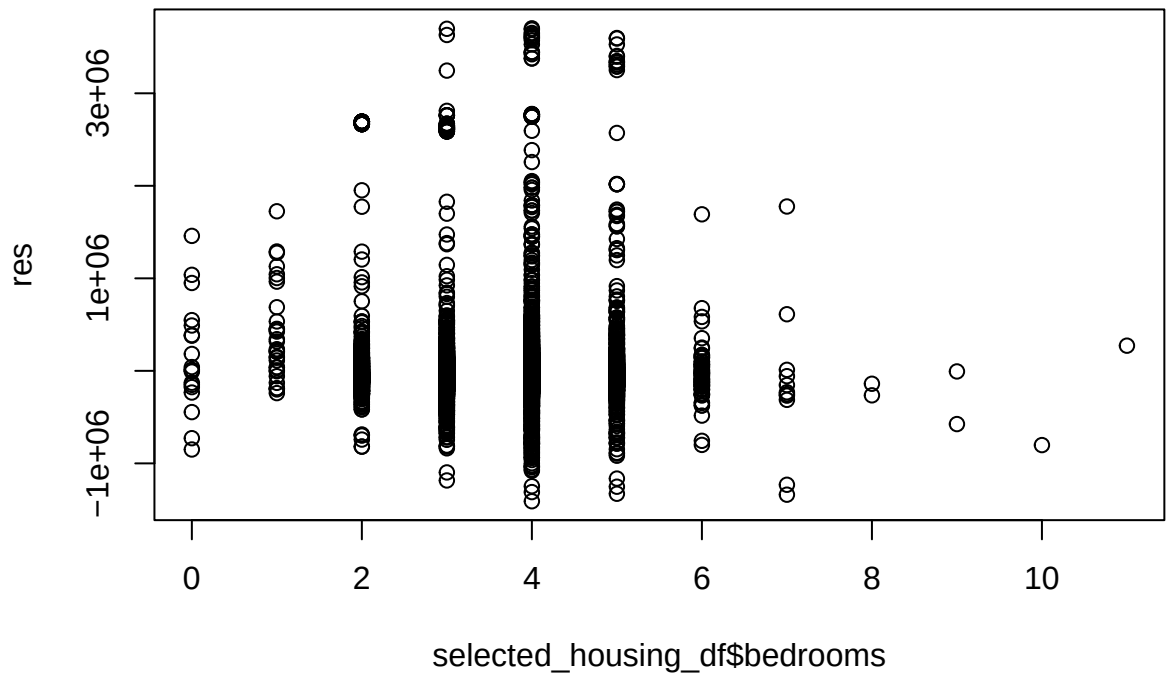

```
plot(selected_housing_df$sq_ft_lot, res)
```



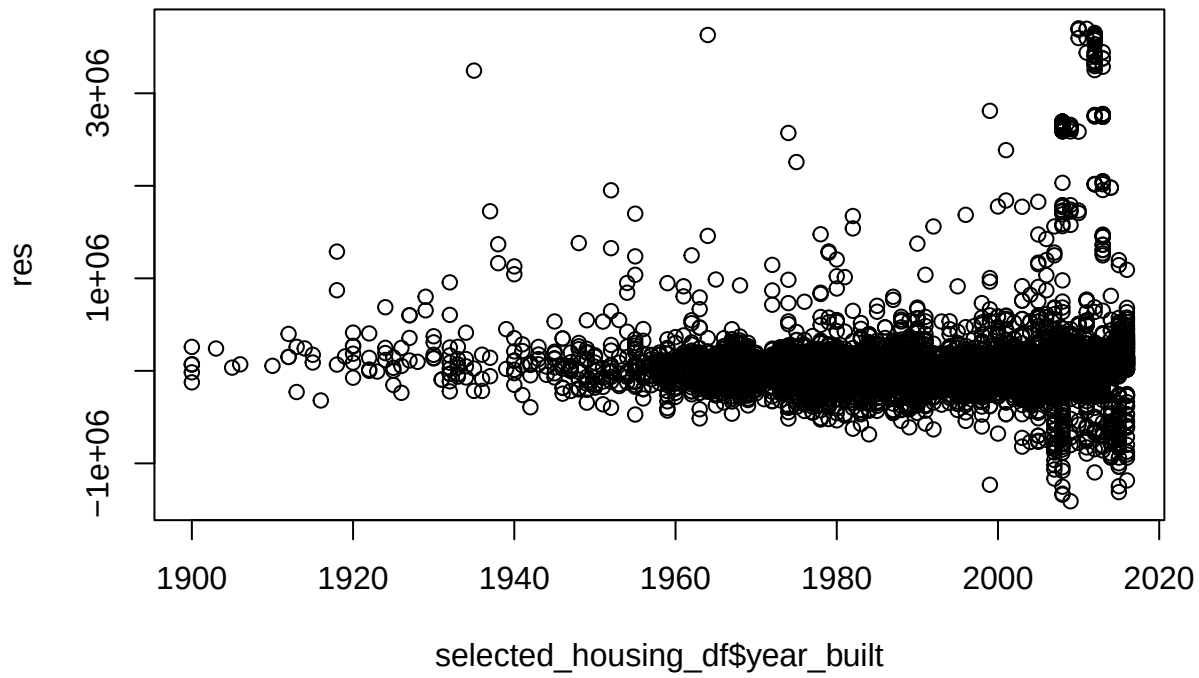

```
plot(selected_housing_df$square_feet_total_living, res)
```



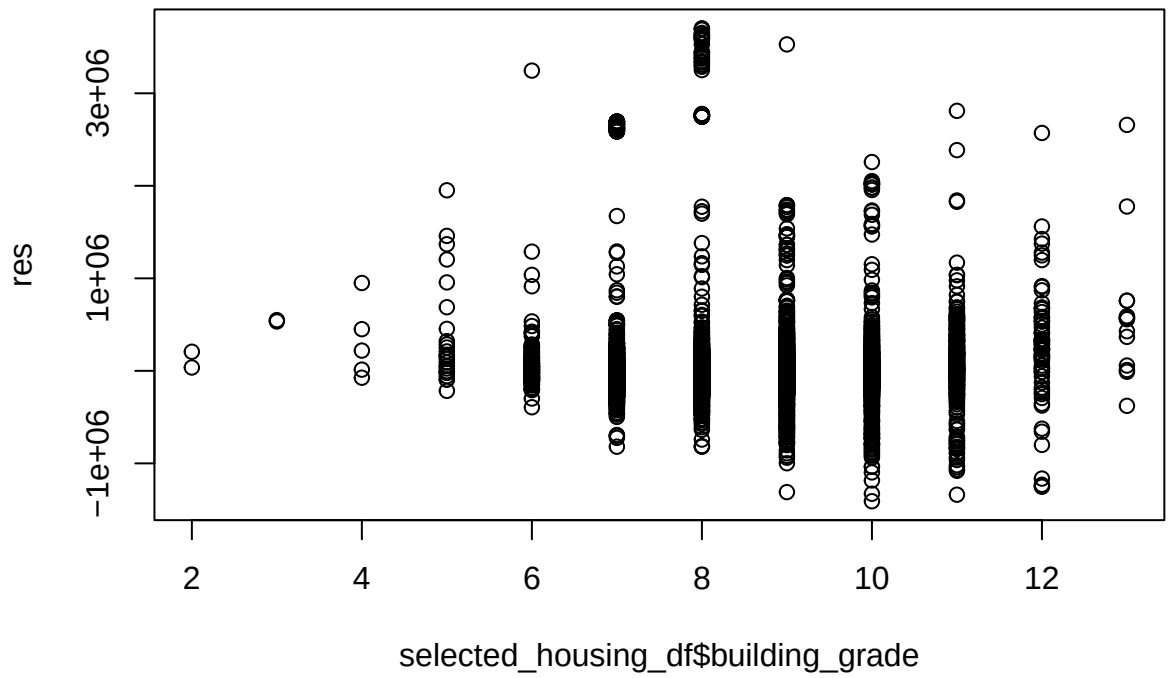

```
plot(selected_housing_df$bedrooms, res)
```




```
plot(selected_housing_df$year_built, res)
```




```
plot(selected_housing_df$building_grade, res)
```




```
# calculating cook's distance, leverage, and covariance rations
cooks_distance <- cooks.distance(multi_lm)
leverage <- hatvalues(multi_lm)
covariance_rations <- covratio(multi_lm)

# creating dataframe with all the new columns in addition to teh standardized
# residuals

library(tibble)
diagnostics_df <- selected_res_df %>% add_column(cooks_distance, leverage, covariance_rations)
head(diagnostics_df)
```

```
##   Sale.Price square_feet_total_living bedrooms sq_ft_lot year_built
## 1     698000                     2810        4      6635       2003
## 2     649990                     2880        4      5570       2006
## 3     572500                     2770        4      8444       1987
## 4     420000                     1620        3      9600       1968
## 5     369900                     1440        3      7526       1980
## 6     184667                     4160        4      7280       2005
##   building_grade     std_res cooks_distance     leverage covariance_rations
## 1              9 -0.12559062   4.782877e-07 0.0001819060           1.000644
## 2              9 -0.31151074   3.019347e-06 0.0001866540           1.000611
## 3              8 -0.26286734   1.641792e-06 0.0001425392           1.000580
## 4              8 -0.09114736   4.875150e-07 0.0003519640           1.000818
## 5              7 -0.16414591   9.115363e-07 0.0002029443           1.000660
## 6              7 -1.98603616   9.544000e-04 0.0014496972           1.000068
```

```r
# interpreting any wild cases

# large cooks distance
lg_cooks <- subset(diagnostics_df, diagnostics_df$cooks_distance > 1)
length(lg_cooks$Sale.Price)
```

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

```r
# there are no values with a cook_distance greater than 1

# average leverage = k+1/n = 6/12786
boundary_avg_leverage <- 3 * 6/12786
large_lev <- subset(diagnostics_df, diagnostics_df$leverage > boundary_avg_leverage)
length(large_lev$Sale.Price)
```

```
## [1] 447
```

```r
# there are 447 values with leverages above the upper bound, basically outliers

upper_cvr <- 1 + 3*6/12786
lower_cvr <- 1 - 3*6/12786

cvr <- subset(diagnostics_df, diagnostics_df$covariance_rations > upper_cvr |
                diagnostics_df$covariance_rations < lower_cvr)
length(cvr$Sale.Price)
```

```
## [1] 781
```

```r
# there are 781 data points that are outide of the boundaries of covariance rations

## xii) assumption of independence test
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
dwt(multi_lm)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7429834     0.5140229       0
##  Alternative hypothesis: rho != 0
```

```r
# the d-value is less than 1.5 which indicates positive autocorrelation,
# and therefore the errors are not independent, which is a cause for concern.

## xiii) multi-collinearity test

# variance inflation factor
vif(multi_lm)
```

```
##                sq_ft_lot square_feet_total_living                 bedrooms
##                 1.148547                 3.349126                 1.634673
##               year_built           building_grade
##                 1.274257                 2.387278
```

```
# the only two variables that have a vif close to one are sq_ft_lot and year_built
# building_grade and square_feet_total_living have higher inflation factors,
# which could indicate moderate correlation, but nothing is higher than 5

# tolerance
1/vif(multi_lm)
```

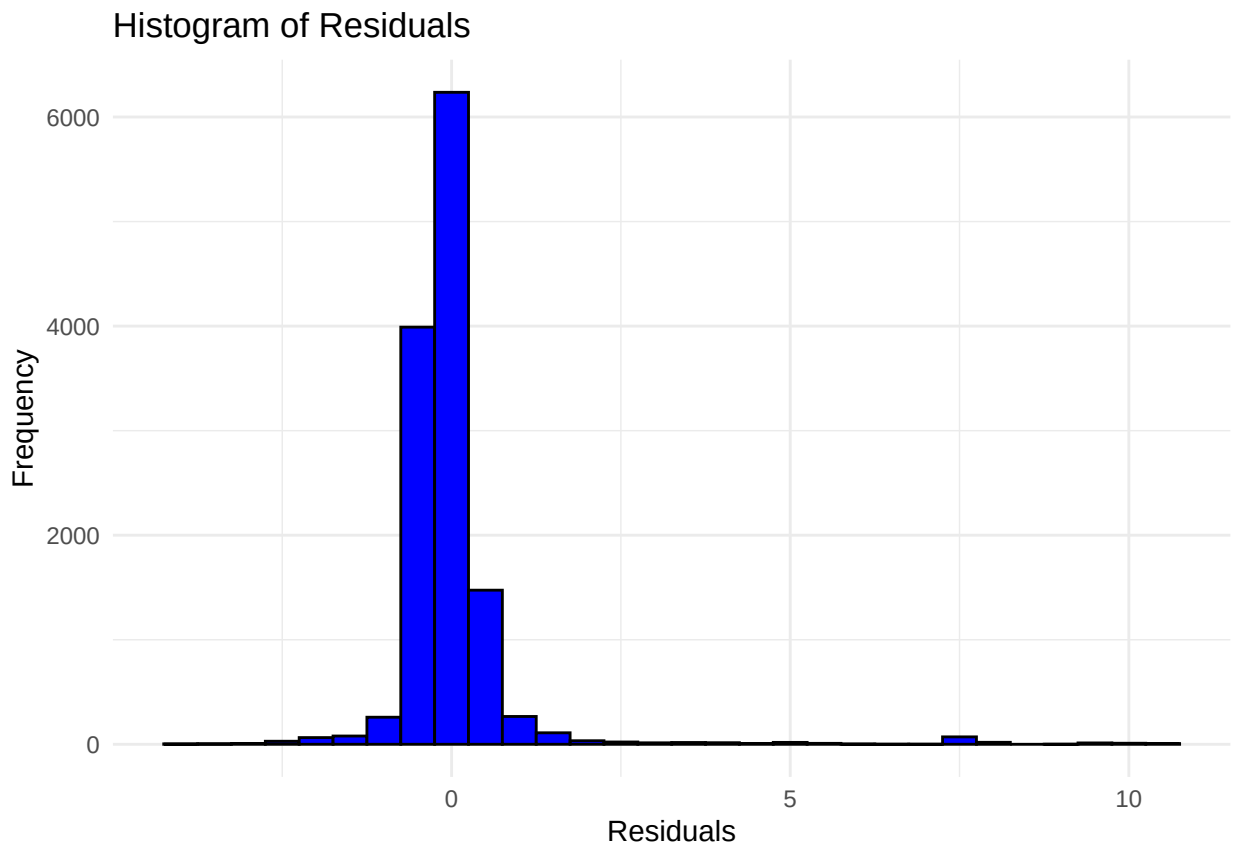
```
##                sq_ft_lot square_feet_total_living                 bedrooms
##                0.8706656                0.2985853                0.6117432
##               year_built           building_grade
##                0.7847713                0.4188871
```

```
# there are no tolerance below 0.25

# average vif
mean(vif(multi_lm))
```

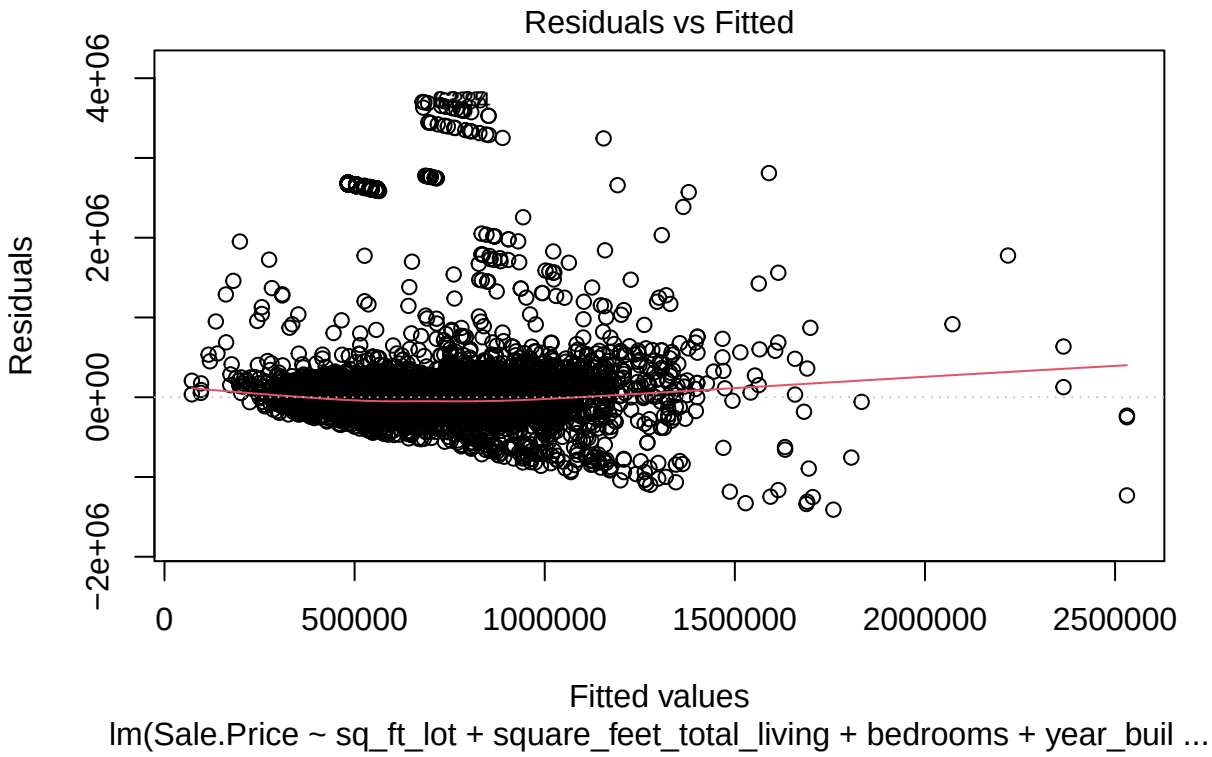
```
## [1] 1.958776
```

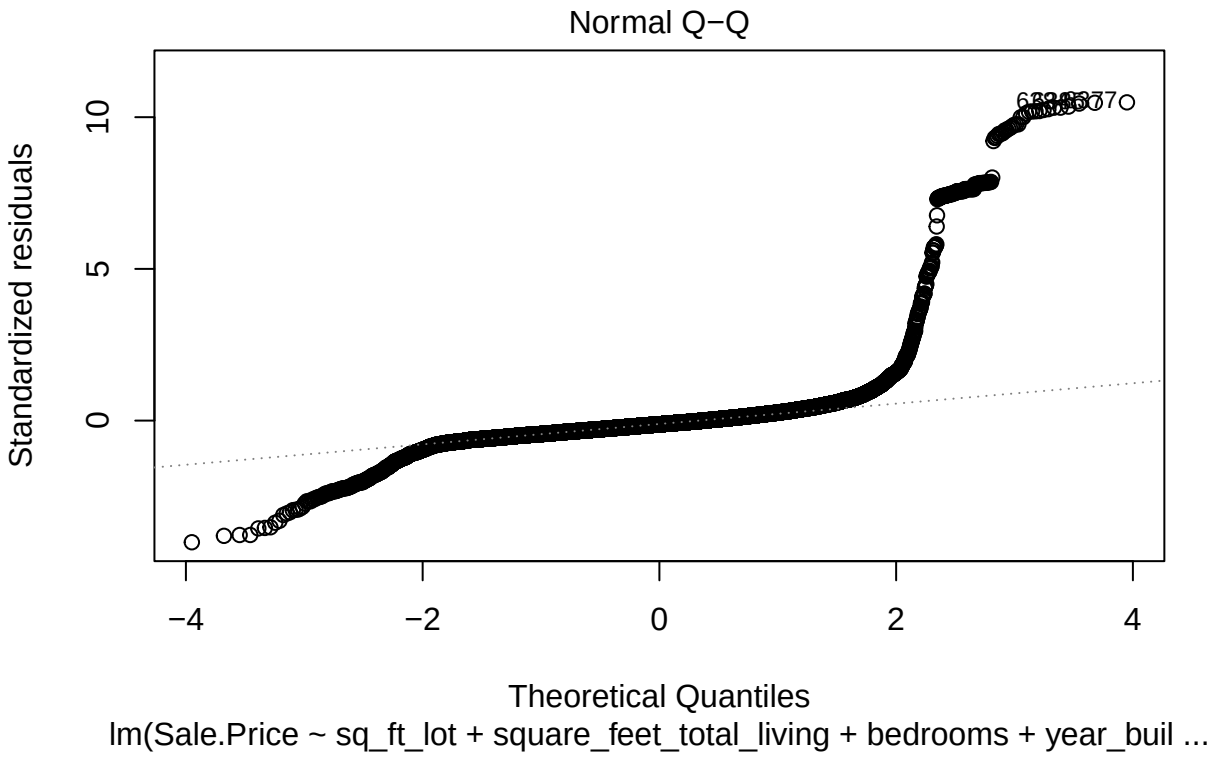
```
# average vif is 1.95 which is pretty close to 1

# so by the tests here, I conclude there isn't any collinearity with our data.

## xiv) visual residuals

# so I did plot the residual plots above comparing each variable to the residuals
# but I can also plot the residuals on a histogram
library(ggplot2)
ggplot(diagnostics_df, aes(x=diagnostics_df$std_res)) +
  geom_histogram(fill = 'blue', color = 'black') +
  labs(title = 'Histogram of Residuals', x = 'Residuals', y = 'Frequency')
```

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

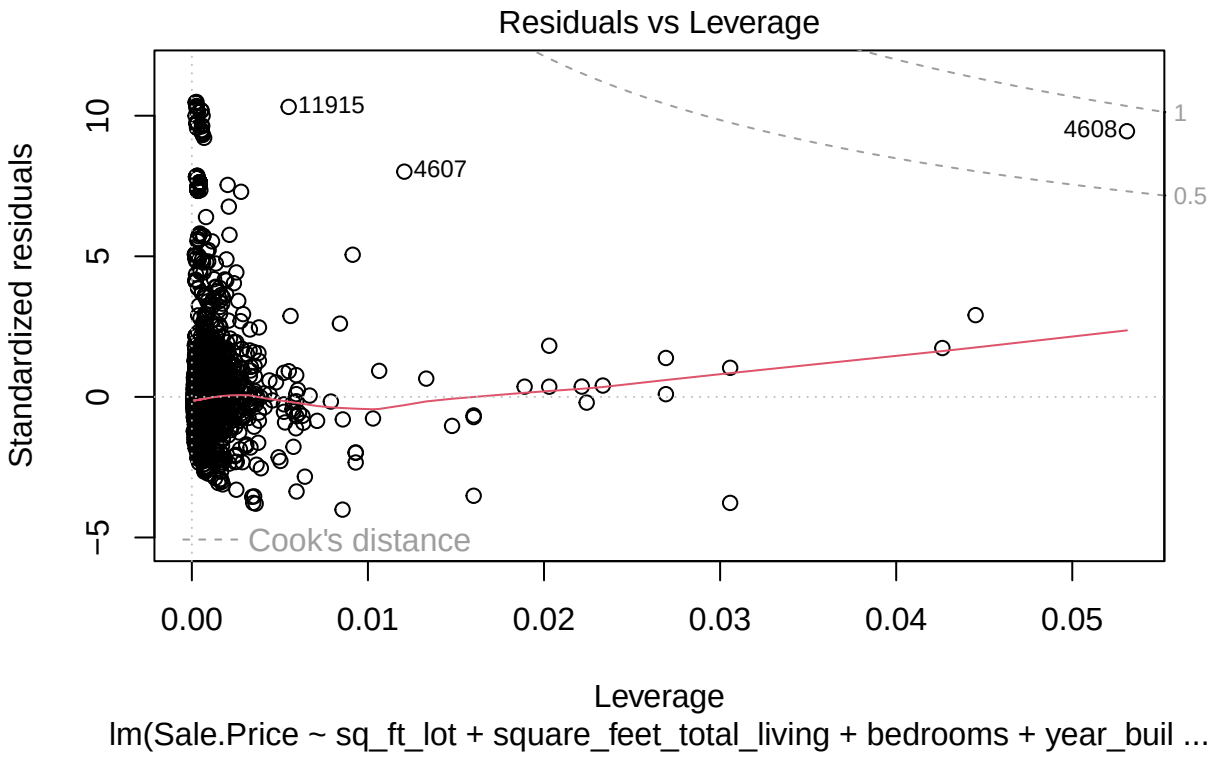



```
plot(multi_lm)
```


Residuals vs Fitted
Residuals
4e+06
2e+06
0e+00
−2e+06
0
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bedrooms + year_buil ...


Normal Q−Q
Standardized residuals
0
5
10
−4
−2
0
2
4
Theoretical Quantiles
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bedrooms + year_buil ...


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bedrooms + year_buil ...




```
# most of the residuals are in the 0-2 range, but there are many anomalies outside
# of the main distribution.From the residuals and fitted graph, the plot doesn't
# behave well, it doesn't form a tight horizontal band after a certain point
# and the plot appears to be parabolic in nature
# the qq plot follows a tight band as well up to a certain point
# the scale-location graph is also kind of sad, it doesn't have equal variability
# at all fitted values

## xv) checking for bias using bootstrapping
library(boot)
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:car':
##
##     logit
```

```
# boot function
bootReg <- function(formula, data, i)
{
  d<- data[i,]
  fit <- lm(formula, data=d)
  return(coef(fit))
}

# boot results
```

```
bootResults<-boot(statistic = bootReg, formula = Sale.Price ~ sq_ft_lot + square_feet_total_living +
                    bedrooms + year_built + building_grade, data = selected_housing_df, R=100,)

# checking the difference between model confint and boot conf int
bootResults
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = selected_housing_df, statistic = bootReg, R = 100,
##     formula = Sale.Price ~ sq_ft_lot + square_feet_total_living +
##         bedrooms + year_built + building_grade)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -5.294444e+06  1.113973e+05 5.661752e+05
## t2*  5.581313e-01 -3.666820e-02 1.678063e-01
## t3*  1.449787e+02  2.398951e+00 7.688033e+00
## t4* -3.982097e+03 -1.304904e+03 5.740095e+03
## t5*  2.684930e+03 -5.695413e+01 2.917077e+02
## t6*  2.929900e+04  1.461038e+02 4.962710e+03
```

```
# boot confidence intervals, I don't know how to combine results into a nice
# data frame
boot.ci(boot.out = bootResults, type = 'norm', index= 1)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 1)
##
## Intervals :
## Level      Normal
## 95%   (-6515524, -4296158 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out = bootResults, type = 'norm', index= 2)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 2)
##
## Intervals :
## Level      Normal
## 95%   ( 0.2659,  0.9237 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out = bootResults, type = 'norm', index= 3)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
```

```
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 3)
##
## Intervals :
## Level      Normal
## 95%   (127.5, 157.6 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out = bootResults, type = 'norm', index= 4)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 4)
##
## Intervals :
## Level      Normal
## 95%   (-13928,   8573 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out = bootResults, type = 'norm', index= 5)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 5)
##
## Intervals :
## Level      Normal
## 95%   (2170, 3314 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out = bootResults, type = 'norm', index= 6)
```

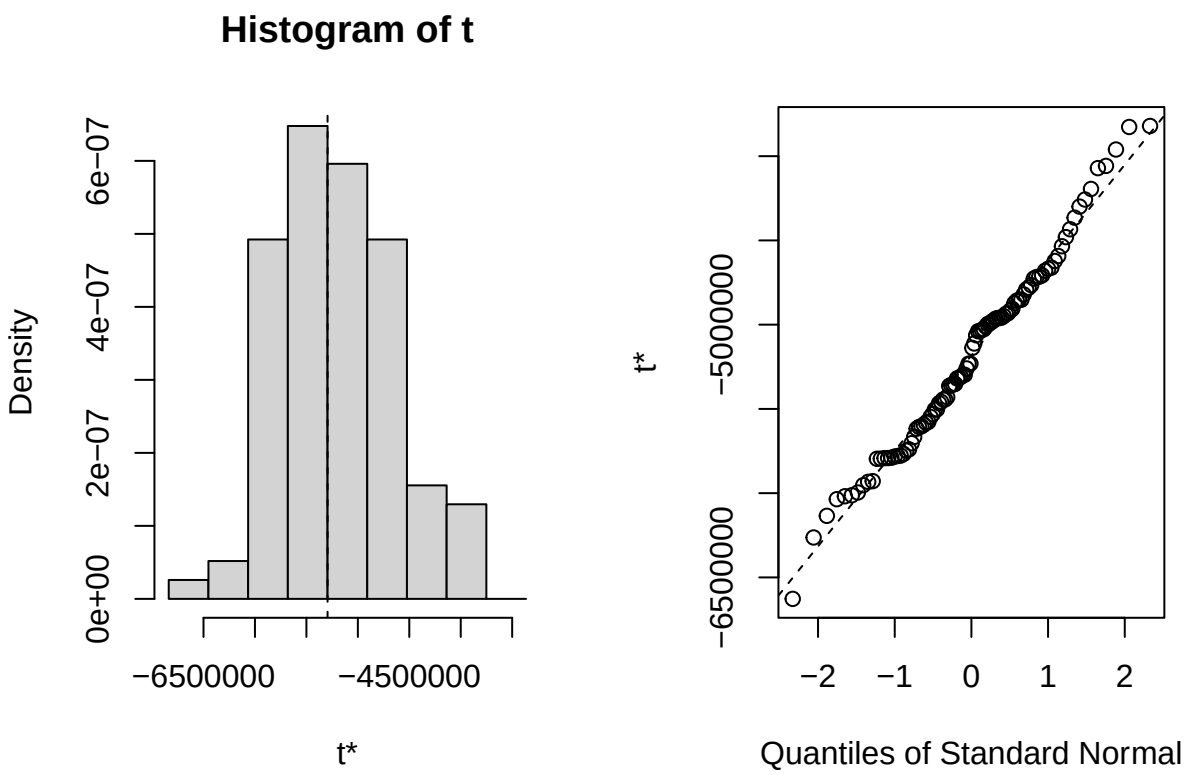
```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootResults, type = "norm", index = 6)
##
## Intervals :
## Level      Normal
## 95%   (19426, 38880 )
## Calculations and Intervals on Original Scale
```

```
# confidence intervals of the model
confint(multi_lm)
```

```
##                                 2.5 %        97.5 %
## (Intercept)              -6.088795e+06 -4.500093e+06
## sq_ft_lot                 4.329206e-01  6.833419e-01
## square_feet_total_living  1.336299e+02  1.563275e+02
## bedrooms                 -1.291387e+04  4.949675e+03
```

```
## year_built                2.282271e+03  3.087588e+03
## building_grade            2.062483e+04  3.797316e+04
```

```
# plotting the bootstrap results, very normalized and standardized
plot(bootResults)
```




```
# the confidence intervals for the beta coefficients
# of the bootstrap model and the normal model are not very close which could
# demonstrate that there could be some bias in the model. The intervals for
# the intercept, bedrooms, year built, and building grade are closer in the
# bootstrap to the original model, and the square_ft_lot and square_feet_total_living
# confidence intervals are tighter in the model than in the boostrapped.

# conclusion: there could be slight bias due to the difference in confidence
# intervals
```