ASSIGNMENT 8.2.3.b

Kiran Komati

2021-05-13

```
knitr::opts_chunk$set(echo = FALSE)
knitr::opts_knit$set(root.dir = 'C:/Users/kiran/dsc520')
```

Load the necessary libraries and load the housing data

```
library("tidyverse")
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## Warning: package 'purrr' was built under R version 4.0.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("readxl")
library(dplyr)
HDF <- read_excel('data/week-6-housing.xlsx')
```

Complete the following:

Explain any transformations or modifications you made to the dataset

```
HDF = rename(HDF,sale_price = `Sale Price`, sale_date=`Sale Date`)
HDFSS <- separate(HDF, sale_date, c("sale_year", "sale_month", "sale_day"), sep = "-") %>%
    dplyr::select(sale_price,sq_ft_lot,square_feet_total_living,bedrooms,bath_full_count,year_built,buil
HDFSS$sale_year <- as.numeric(as.character(HDFSS$sale_year))
```

1. Renamed `Sale Price` to sale_price and `Sale Date` to sale_date.
2. Extracted "sale_year", "sale_month", "sale_day" from the field "sale_date".
3. Converted "sale_year" to numeric.
4. Created a subset with sale_price,sq_ft_lot,square_feet_total_living,bedrooms,bath_full_count,year_built,building_g fields in it.

Create two variables; one that will contain the variables Sale Price and Square Foot of Lot (same variables used from previous assignment on simple regression) and one that will contain Sale Price and several additional predictors of your choice. Explain the basis for your additional predictor selections.

```
HDF_lm <-  lm(sale_price ~ sq_ft_lot,data = HDF)
HDFSS_lm <-  lm(sale_price ~ sq_ft_lot + square_feet_total_living + +building_grade  + bath_full_count +
```

Used correlation to identify the variables that are related to the sale price and picked those that have relatively strong correlation.

Execute a summary() function on two variables defined in the previous step to compare the model results. What are the R2 and Adjusted R2 statistics? Explain what these results tell you about the overall model. Did the inclusion of the additional predictors help explain any large variations found in Sale Price?

```
summary(HDF_lm)
```

```
##
## Call:
## lm(formula = sale_price ~ sq_ft_lot, data = HDF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2016064  -194842   -63293    91565  3735109
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.418e+05  3.800e+03  168.90   <2e-16 ***
## sq_ft_lot   8.510e-01  6.217e-02   13.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 401500 on 12863 degrees of freedom
## Multiple R-squared:  0.01435,    Adjusted R-squared:  0.01428
## F-statistic: 187.3 on 1 and 12863 DF,  p-value: < 2.2e-16
```

```r
summary(HDFSS_lm)
```

```
##
## Call:
## lm(formula = sale_price ~ sq_ft_lot + square_feet_total_living +
##     +building_grade + bath_full_count + year_built + bedrooms,
##     data = HDF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2103184  -118385   -41919    42901  3743675
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -4.693e+06  4.321e+05 -10.861  < 2e-16 ***
## sq_ft_lot                 2.842e-01  5.907e-02   4.812 1.51e-06 ***
## square_feet_total_living  1.418e+02  5.926e+00  23.931  < 2e-16 ***
## building_grade            3.131e+04  4.456e+03   7.026 2.23e-12 ***
## bath_full_count           1.412e+04  6.094e+03   2.317   0.0205 *
## year_built                2.371e+03  2.195e+02  10.803  < 2e-16 ***
## bedrooms                 -6.137e+03  4.606e+03  -1.332   0.1827
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 356400 on 12858 degrees of freedom
## Multiple R-squared:  0.2237, Adjusted R-squared:  0.2233
## F-statistic: 617.4 on 6 and 12858 DF,  p-value: < 2.2e-16
```

R2 and adjusted R2 increased after adding the new predictors to the linear model which means the multiple linear regression with the selected variables did a relatively good job in predicting prices and accounts for nearly 22% of the values.

Considering the parameters of the multiple regression model you have created. What are the standardized betas for each parameter and what do the values indicate?

```r
library('QuantPsyc')
```

```
## Warning: package 'QuantPsyc' was built under R version 4.0.5
```

```
## Loading required package: boot
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

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

##
## Attaching package: 'QuantPsyc'

## The following object is masked from 'package:base':
##
##     norm
```

```
lm.beta(HDFSS_lm)
```

```
##               sq_ft_lot square_feet_total_living           building_grade
##              0.04001884               0.34713398               0.08458526
##         bath_full_count               year_built                 bedrooms
##              0.02272025               0.10096737              -0.01329565
```

These estimates tell us the number of standard deviations by which the outcome will change as a result of one standard deviation change in the predictor.

Calculate the confidence intervals for the parameters in your model and explain what the results indicate.

```
confint(HDFSS_lm)
```

```
##                                 2.5 %        97.5 %
## (Intercept)              -5.540197e+06 -3.846162e+06
## sq_ft_lot                 1.684509e-01  4.000341e-01
## square_feet_total_living  1.302023e+02  1.534347e+02
## building_grade            2.257129e+04  4.003889e+04
## bath_full_count           2.173236e+03  2.606182e+04
## year_built                1.940810e+03  2.801222e+03
## bedrooms                 -1.516439e+04  2.891031e+03
```

This confidence interval tells us that the predictors (sq ft living total,bath_full_count) have very tight confidence intervals, indicating that the estimates for the current model are likely to be representative of the true population values. The interval for sq_ft_lot,building_grade,bath_full_count,year_built is wider (but still does not cross zero), indicating that the parameter for this variable is less representative, but nevertheless significant.For predictor "bedrooms" the value crossed zero indicating that in some samples the predictor has a negative relationship to the outcome whereas in others it has a positive relationship.

Assess the improvement of the new model compared to your original model (simple regression model) by testing whether this change is significant by performing an analysis of variance.

```r
anova(HDF_lm,HDFSS_lm)
```

```
## Analysis of Variance Table
##
## Model 1: sale_price ~ sq_ft_lot
## Model 2: sale_price ~ sq_ft_lot + square_feet_total_living + +building_grade +
##     bath_full_count + year_built + bedrooms
##   Res.Df        RSS Df Sum of Sq      F    Pr(>F)
## 1  12863 2.0734e+15
## 2  12858 1.6331e+15  5 4.403e+14 693.34 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The value of F is 693.34, we can say that multiple regression significantly improved the fit of the model to the data compared to simple regression, $F(5, 12858) = 693.34$, $p < .001$.

Perform casewise diagnostics to identify outliers and/or influential cases, storing each function's output in a dataframe assigned to a unique variable name.

```r
HDFSS$standardized.residuals<- rstandard(HDFSS_lm)
HDFSS$studentized.residuals<-rstudent(HDFSS_lm)
HDFSS$cooks.distance<-cooks.distance(HDFSS_lm)
HDFSS$dfbeta<-dfbeta(HDFSS_lm)
HDFSS$dffit<-dffits(HDFSS_lm)
HDFSS$leverage<-hatvalues(HDFSS_lm)
HDFSS$covariance.ratios<-covratio(HDFSS_lm)
```

Calculate the standardized residuals using the appropriate command, specifying those that are +-2, storing the results of large residuals in a variable you create.

```r
HDFSS$large.residuals <- HDFSS$standardized.residuals > 2 | HDFSS$standardized.residuals < -2
```

Use the appropriate function to show the sum of large residuals.

```r
sum(HDFSS$large.residuals)
```

```
## [1] 335
```

Which specific variables have large residuals (only cases that evaluate as TRUE)?

```
HDFSS[HDFSS$large.residuals,c("sale_price","sq_ft_lot","square_feet_total_living","bedrooms","bath_ful
```

```
## # A tibble: 335 x 8
##    sale_price sq_ft_lot square_feet_total_l~ bedrooms bath_full_count year_built
##         <dbl>     <dbl>                <dbl>    <dbl>           <dbl>      <dbl>
##  1     265000    112650                 4920        4               4       2007
##  2    1390000    225640                  660        0               1       1955
##  3     229000    236966                 3840        0               0       2008
##  4     390000     63162                 5800        5               4       2008
##  5    1588359      8752                 3360        2               2       2005
##  6    1450000     14043                  900        2               1       1918
##  7     163000     18498                 4710        4               2       2014
##  8     270000     89734                 5060        4              23       2016
##  9     200000    288367                 6880        5               1       2008
## 10     300000     55303                 4490        4               2       2008
## # ... with 325 more rows, and 2 more variables: building_grade <dbl>,
## #   standardized.residuals <dbl>
```

Investigate further by calculating the leverage, cooks distance, and covariance rations. Comment on all cases that are problematics.

```
HDFSS[HDFSS$large.residuals , c("cooks.distance", "leverage", "covariance.ratios")]
```

```
## # A tibble: 335 x 3
##    cooks.distance leverage covariance.ratios
##             <dbl>    <dbl>             <dbl>
##  1       0.00111  0.00128              0.999
##  2       0.00368  0.00275              0.998
##  3       0.00314  0.00475              1.00
##  4       0.00116  0.00130              0.998
##  5       0.000465 0.000733             0.999
##  6       0.00376  0.00205              0.996
##  7       0.000741 0.000854             0.998
##  8       0.254    0.121                1.13
##  9       0.00745  0.00439              0.999
## 10       0.000450 0.000677             0.999
## # ... with 325 more rows
```

```
HDFSS %>%
    filter(cooks.distance >1)
```

```
## # A tibble: 0 x 16
## # ... with 16 variables: sale_price <dbl>, sq_ft_lot <dbl>,
## #   square_feet_total_living <dbl>, bedrooms <dbl>, bath_full_count <dbl>,
```

```
## #   year_built <dbl>, building_grade <dbl>, sale_year <dbl>,
## #   standardized.residuals <dbl>, studentized.residuals <dbl>,
## #   cooks.distance <dbl>, dfbeta <dbl[,7]>, dffit <dbl>, leverage <dbl>,
## #   covariance.ratios <dbl>, large.residuals <lgl>
```

None of them has a Cook's distance greater than 1. Values greater than 1 can be considered as a concern.

```
k <- ncol(HDF)
n <- nrow(HDF)
avg <- (k+1)/n

HDFSS %>%
    filter(leverage >3*avg)
```

```
## # A tibble: 41 x 16
##    sale_price sq_ft_lot square_feet_total_l~ bedrooms bath_full_count year_built
##         <dbl>     <dbl>                <dbl>    <dbl>           <dbl>      <dbl>
##  1    1490000    871202                 3540        4               2       1999
##  2     270000     89734                 5060        4              23       2016
##  3     275000    532739                 2550        2               2       2013
##  4      90000    574992                 2700        3               1       2003
##  5      90000    574992                 1380        3               1       2003
##  6    1299950    669952                 2800        3               2       1987
##  7      32000    544199                 4740        6               5       2007
##  8     349999     45738                 9360        4               3       2009
##  9    2988000    207781                10630        5               4       2003
## 10    1825000     13220                 7980       11               3       2002
## # ... with 31 more rows, and 10 more variables: building_grade <dbl>,
## #   sale_year <dbl>, standardized.residuals <dbl>, studentized.residuals <dbl>,
## #   cooks.distance <dbl>, dfbeta <dbl[,7]>, dffit <dbl>, leverage <dbl>,
## #   covariance.ratios <dbl>, large.residuals <lgl>
```

Around 41 cases where leverage greater than three times average which can unduly influence the model.

```
CVR_Upperlimit<-1+(3*(k+1)/n)
CVR_Lowerlimit<-1-(3*(k+1)/n)

HDFSS %>%
    filter(covariance.ratios> CVR_Upperlimit | covariance.ratios< CVR_Lowerlimit) %>%
    dplyr::select(covariance.ratios,cooks.distance)
```

```
## # A tibble: 212 x 2
##    covariance.ratios cooks.distance
##                <dbl>          <dbl>
##  1              1.02        0.00326
##  2              1.13        0.254
##  3              1.01        0.00275
##  4              1.01        0.00544
##  5              1.01        0.00295
##  6              0.991       0.00190
##  7              0.987       0.00403
##  8              1.01        0.00209
```

```
##  9              0.988       0.00165
## 10              0.987       0.000711
## # ... with 202 more rows
```

Covariance ratio. There are 212 records where the covariance ratio is not with in the boundaries. But none of them are way beyond the limits which means that there are no significant cases that influence the model.

Perform the necessary calculations to assess the assumption of independence and state if the condition is met or not.

```r
library('car')
```

```
## Warning: package 'car' was built under R version 4.0.5

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:boot':
##
##     logit

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

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

```r
durbinWatsonTest(HDFSS_lm)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7294196     0.5411509       0
##  Alternative hypothesis: rho != 0
```

Here the value is 0.541 which is less than 1 and it means that the assumption of independence is not met.

Perform the necessary calculations to assess the assumption of no multicollinearity and state if the condition is met or not.

```r
vif(HDFSS_lm)
```

```
##                sq_ft_lot square_feet_total_living           building_grade
##                 1.145650                 3.485015                 2.400560
##          bath_full_count               year_built                 bedrooms
##                 1.592856                 1.446746                 1.649129
```

```
1/vif(HDFSS_lm)
```

```
##                sq_ft_lot square_feet_total_living           building_grade
##                0.8728672                0.2869428                0.4165695
##          bath_full_count               year_built                 bedrooms
##                0.6278031                0.6912063                0.6063809
```

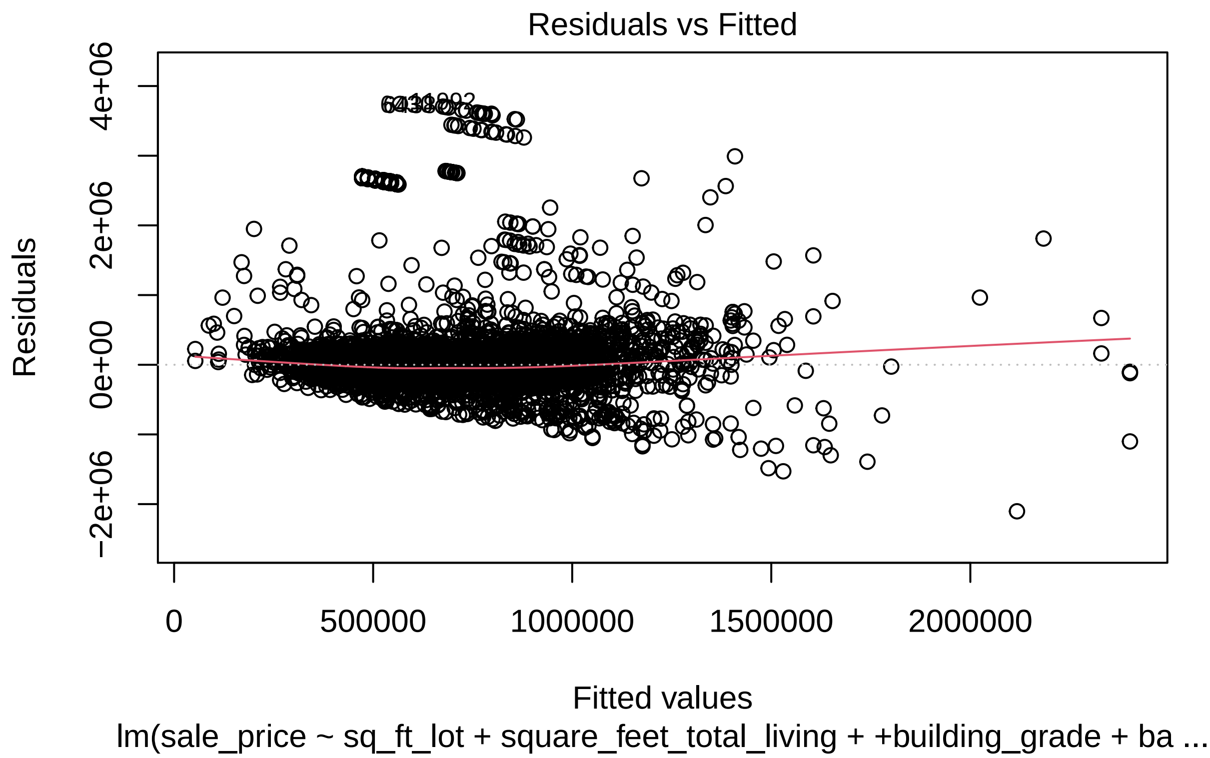
```
mean(vif(HDFSS_lm))
```

```
## [1] 1.953326
```

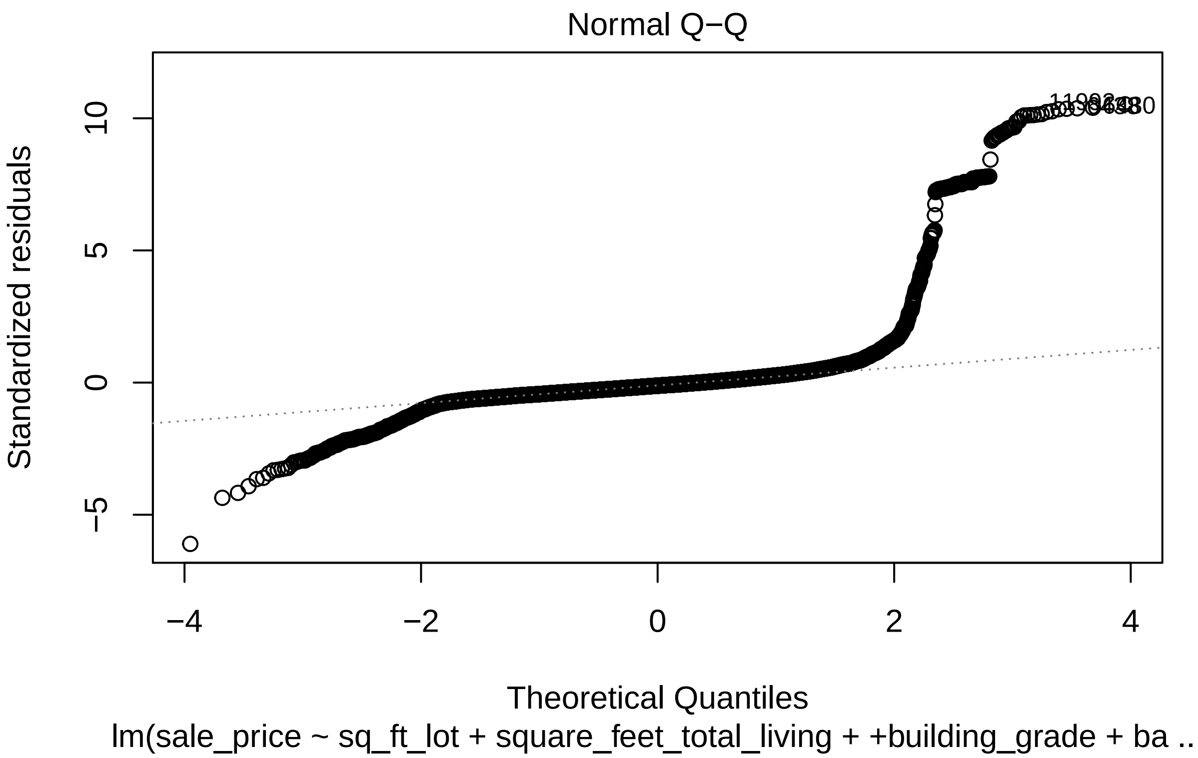
All the VIF values are well below 10 and the tolerance statistics are well above 0.2 , the mean VIF is 1.953. we can safely conclude that there is no collinearity within our data.

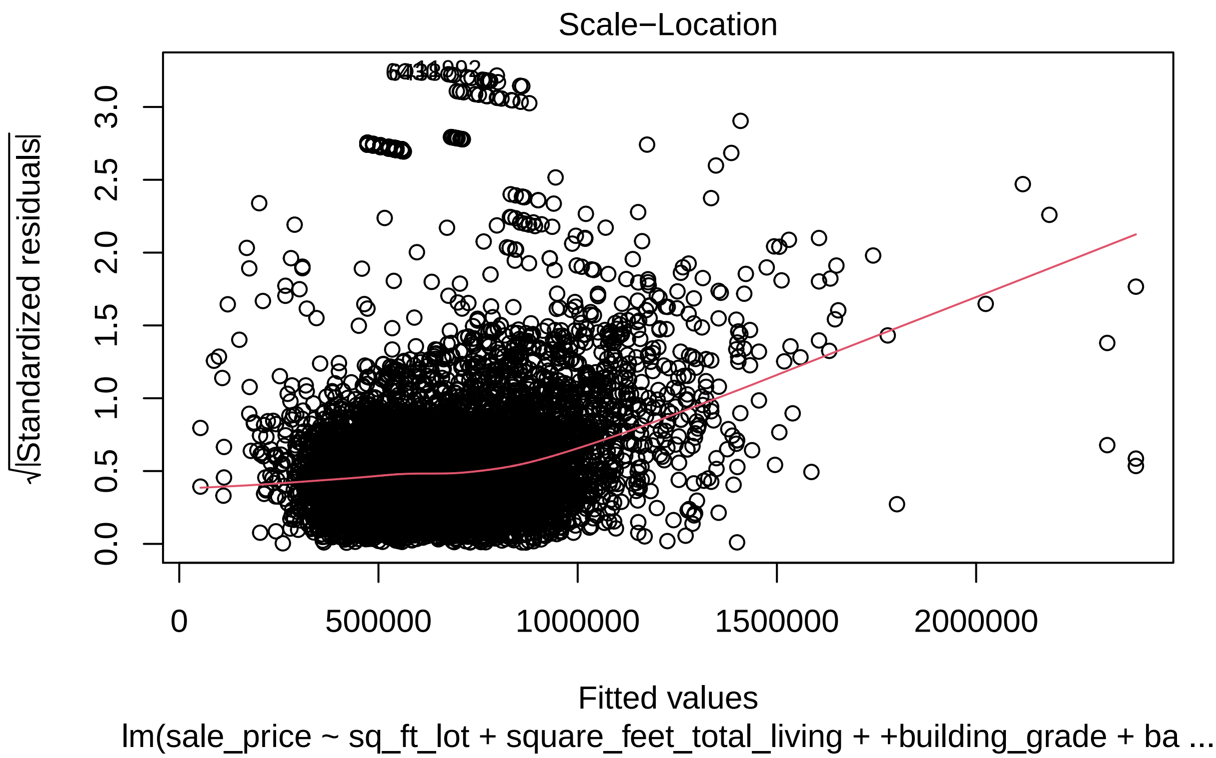
Visually check the assumptions related to the residuals using the plot() and hist() functions. Summarize what each graph is informing you of and if any anomalies are present.

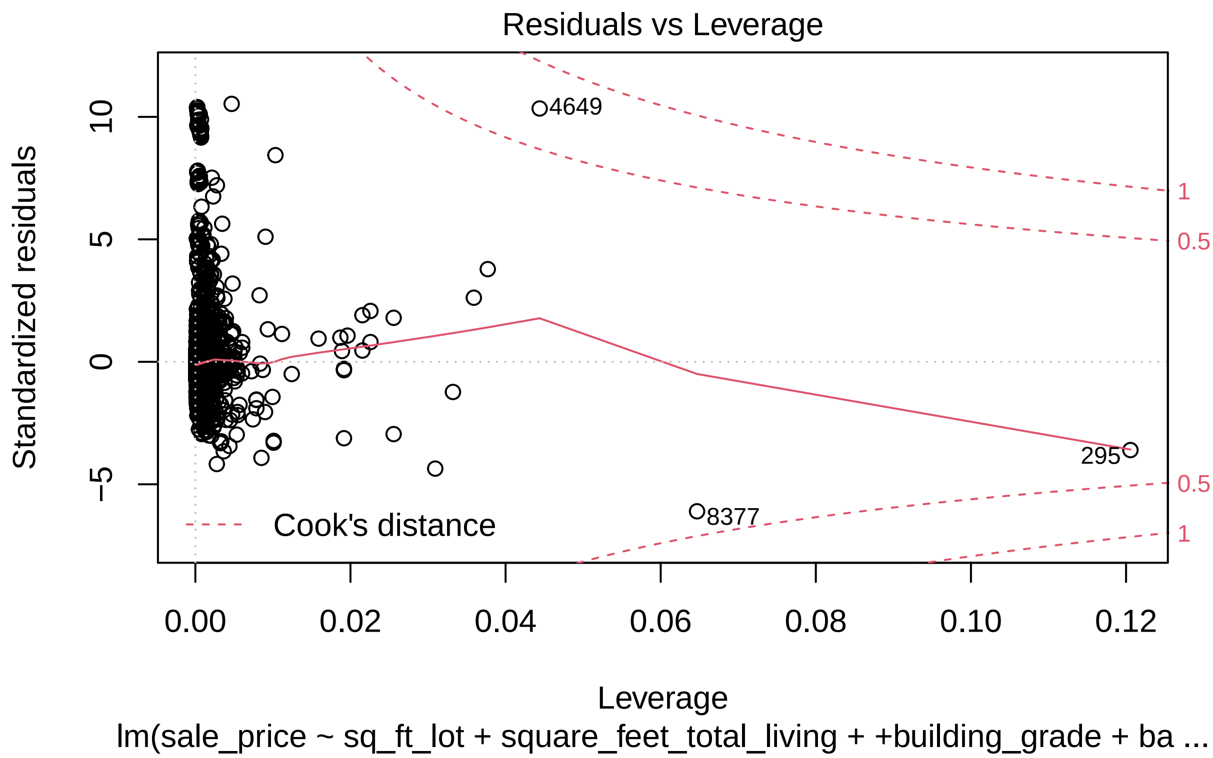
```
library(ggplot2)
plot(HDFSS_lm)
```



Residuals vs Fitted
Residuals
Fitted values
lm(sale_price ~ sq_ft_lot + square_feet_total_living + +building_grade + ba ...


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(sale_price ~ sq_ft_lot + square_feet_total_living + +building_grade + ba ...
10
5
0
−5
−4
−2
0
2
4


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
500000
1000000
1500000
2000000
Fitted values
lm(sale_price ~ sq_ft_lot + square_feet_total_living + +building_grade + ba ...




```
hist(HDFSS$studentized.residuals)
```

The plot function shows that values are evenly distributed around zero which indicates that the assumptions of linearity, randomness and homoscedasticity have been met. The second graph is skewed which shows that there's deviation from normality. Histogram shows that the distribution is not normal and that is is right skewed(assymetrical).

Overall, is this regression model unbiased? If an unbiased regression model, what does this tell us about the sample vs. the entire population model?

For a model to be unbiased, there are several assumptions that must be true. For our model, assumption of independence is not met and errors are not normally distributed. If a model is unbiased, it means that on an average the regression model from sample is same as the population model.