Housing_Data_Assignment

Theodore Koby-Hercsky

5/9/2021

html_document: https://rpubs.com/theoKoby/769360

Set the working directory to the root of your DSC 520 directory

setwd("~/Documents/Bellevue University Classes/DSC520/assignments/assignment06")

The housing.csv file

```
## I am importing readr from the library so I can use the read_csv function t
o create my student survey data frame.
library(readr)
## Creating the student survey data frame by using the read_csv function to p
ull my student survey data.
Housing_df <- read_csv("data/housing.csv")
Housing_df

## # A tibble: 12,865 x 24
##    sale_date sale_price sale_reason sale_instrument sale_warning sitetype
##    <chr>          <dbl>       <dbl>           <dbl> <chr>        <chr>
##  1 1/3/06        698000           1               3 No           R1
##  2 1/3/06        649990           1               3 No           R1
##  3 1/3/06        572500           1               3 No           R1
##  4 1/3/06        420000           1               3 No           R1
##  5 1/3/06        369900           1               3 Yes          R1
##  6 1/3/06        184667           1              15 Yes          R1
##  7 1/4/06       1050000           1               3 No           R1
##  8 1/4/06        875000           1               3 No           R1
##  9 1/4/06        660000           1               3 No           R1
## 10 1/4/06        650000           1               3 No           R1
## # … with 12,855 more rows, and 18 more variables: addr_full <chr>,
## #   zip_code <dbl>, ctyname <chr>, postalctyn <chr>, lon <dbl>, lat <dbl>,
## #   building_grade <dbl>, square_feet_total_living <dbl>, bedrooms <dbl>,
## #   bath_full_count <dbl>, bath_half_count <dbl>, bath_3qtr_count <dbl>,
## #   year_built <dbl>, year_renovated <dbl>, current_zoning <chr>,
## #   sq_ft_lot <dbl>, prop_type <chr>, present_use <dbl>
```

#Explain any transformations or modifications you made to the dataset

I modified the sales warning variable by changing the numbers that are useless to us to being a yes for a sales warning and no if there was no data in that field.

I also inputted the city name if the line was blank in the city name variable by checking the name on the postal city name variable.

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.

```
## I fit a linear model using the `sq_ft_lot` variable as the predictor and `
sale_price` as the outcome

square_foot_price_lm <- lm(sale_price ~ sq_ft_lot, data = Housing_df)

## When we fit our linear model by using our sq_ft_lot variable as the predic
tor and sale_price as the outcome with the Housing_df as our data we see coef
ficients for intercept at 6.418

square_foot_price_lm

##
## Call:
## lm(formula = sale_price ~ sq_ft_lot, data = Housing_df)
##
## Coefficients:
## (Intercept)    sq_ft_lot
##   6.418e+05    8.510e-01

## Seen below is a linear model using the `zip_code`, `square_feet_total_livi
ng`, `bedrooms`, and `bath_full_count' variables as the predictors and `sale_
price` as the outcome

sale_price_lm <-  lm(sale_price ~ zip_code + square_feet_total_living + bedro
oms + bath_full_count, data = Housing_df)

## We fit the linear model by using our variables as the predictor and sale_p
rice as the outcome with the Housing_df as our data we see coefficients for i
ntercept and `zip_code`, `square_feet_total_living`, `bedrooms`, and `bath_fu
ll_count' which is the slope for the predictors.

sale_price_lm

##
## Call:
## lm(formula = sale_price ~ zip_code + square_feet_total_living +
##     bedrooms + bath_full_count, data = Housing_df)
##
```

```
## Coefficients:
##             (Intercept)                  zip_code  square_feet_total_livi
ng
##              -1.999e+08                 2.041e+03                 1.837e+
02
##                bedrooms           bath_full_count
##              -2.466e+04                 4.195e+04
```

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?

```
## I will view the full report by using summary of my square_foot_price_lm mo
del
## As seen below we see that the Multiple R-squared: is 0.01435 and the Adjus
ted R-squared: is 0.01428
## Which is expressed as a percentage between 0 and 100, with 100 signaling p
erfect correlation and zero no correlation at all. As in the summary of the s
quare_foot_price_lm model we find that the Multiple R-squared and Adjusted R-
squared are less than 2% which is very low meaning there is very little corre
lation. Which I find to be quite interesting as you would expect the square f
ootage to have a higher impact on the price.
summary(square_foot_price_lm)

##
## Call:
## lm(formula = sale_price ~ sq_ft_lot, data = Housing_df)
##
## 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

## The Summary for sale_price_lm can been seen when we use the Summary functi
on seen below
## As seen below we see that the Multiple R-squared: is 0.2121 and the Adjust
ed R-squared: is 0.2118
## As for the summary of the sale_price_lm model we find that the Multiple R-
squared and Adjusted R-squared are significantly higher than that of the squa
```

```
re_foot_price_lm meaning there is a higher correlation in comparison to the p
revious test. While in the inclusion of the additional predictors helped expl
ain the variance in sales price as the zip code, bedrooms, and bathrooms have
helped increase the sales price.
summary(sale_price_lm)

##
## Call:
## lm(formula = sale_price ~ zip_code + square_feet_total_living +
##     bedrooms + bath_full_count, data = Housing_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1759242  -117826   -41208    43890  3832749
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -1.999e+08  1.854e+08  -1.078    0.281
## zip_code                   2.041e+03  1.891e+03   1.079    0.281
## square_feet_total_living   1.837e+02  4.377e+00  41.964  < 2e-16 ***
## bedrooms                  -2.466e+04  4.446e+03  -5.547 2.96e-08 ***
## bath_full_count            4.195e+04  5.695e+03   7.366 1.87e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 359000 on 12860 degrees of freedom
## Multiple R-squared:  0.2121, Adjusted R-squared:  0.2118
## F-statistic: 865.3 on 4 and 12860 DF,  p-value: < 2.2e-16
```

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?

```
install.packages("car")

##
## The downloaded binary packages are in
##  /var/folders/wm/_x82v16s45bgfptwshv23l3r0000gp/T//RtmpNkjeiI/downloaded_p
ackages

library(car)
## I used the comparecoefs to dive deeper into this question and found that t
he the most significant difference lied within the bedrooms and full baths.
compareCoefs(square_foot_price_lm, sale_price_lm)

## Calls:
## 1: lm(formula = sale_price ~ sq_ft_lot, data = Housing_df)
## 2: lm(formula = sale_price ~ zip_code + square_feet_total_living + bedroom
s
##   + bath_full_count, data = Housing_df)
##
```

```
##                              Model 1   Model 2
## (Intercept)                 6.42e+05 -2.00e+08
## SE                          3.80e+03  1.85e+08
##                                                
## sq_ft_lot                     0.8510          
## SE                            0.0622          
##                                                
## zip_code                                  2041
## SE                                        1891
##                                                
## square_feet_total_living                183.66
## SE                                        4.38
##                                                
## bedrooms                                -24663
## SE                                        4446
##                                                
## bath_full_count                          41949
## SE                                        5695
##
```

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

```
library(MASS, pos = 15)
## This shows us a 95% confidence intervals of 631569.6 645448.7 for the para
meter sq_ft_lot.
with(Housing_df, (t.test(sale_price, sq_ft_lot, alternative = 'two.sided', co
nf.level = .95, paired = TRUE)))
```

```
##
##  Paired t-test
##
## data:  sale_price and sq_ft_lot
## t = 180.35, df = 12864, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  631569.6 645448.7
## sample estimates:
## mean of the differences
##                638509.2
```

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.

```
## I used the compareCoefs again to see if the change is significant and we f
ound that we have a significant difference between model 1 and 2.
compareCoefs(square_foot_price_lm, sale_price_lm)
```

```
## Calls:
## 1: lm(formula = sale_price ~ sq_ft_lot, data = Housing_df)
```

```
## 2: lm(formula = sale_price ~ zip_code + square_feet_total_living + bedroom
s
##   + bath_full_count, data = Housing_df)
##
##                              Model 1   Model 2
## (Intercept)                6.42e+05 -2.00e+08
## SE                         3.80e+03  1.85e+08
##
## sq_ft_lot                    0.8510
## SE                           0.0622
##
## zip_code                                 2041
## SE                                       1891
##
## square_feet_total_living               183.66
## SE                                       4.38
##
## bedrooms                               -24663
## SE                                       4446
##
## bath_full_count                         41949
## SE                                       5695
##
```

```r
## I also created anovas for each model separately to see the differences on
their own.

## In this test we will be performing an analysis of variance on the square_f
oot_price_lm
square_footAnova <- aov(sale_price ~ sq_ft_lot, data = Housing_df)
## We see that the Estimated effects may be unbalanced as the Residual standa
rd error: 401483.8 which shows a significant room for error
square_footAnova
```

```
## Call:
##    aov(formula = sale_price ~ sq_ft_lot, data = Housing_df)
##
## Terms:
##                       sq_ft_lot    Residuals
## Sum of Squares  3.019674e+13 2.073377e+15
## Deg. of Freedom            1        12863
##
## Residual standard error: 401483.8
## Estimated effects may be unbalanced
```

```r
## In this test we will be performing an analysis of variance on the sale_pri
ce_lm
SaleAnova <- aov(sale_price ~ zip_code + square_feet_total_living + bedrooms
+ bath_full_count, data = Housing_df)
## We see that the Estimated effects may be unbalanced as the Residual standa
```

```
rd error: 359008.6
SaleAnova

## Call:
##    aov(formula = sale_price ~ zip_code + square_feet_total_living +
##     bedrooms + bath_full_count, data = Housing_df)
##
## Terms:
##                     zip_code square_feet_total_living     bedrooms
## Sum of Squares  7.610438e+12             4.276822e+14 3.798816e+12
## Deg. of Freedom            1                        1            1
##                 bath_full_count    Residuals
## Sum of Squares     6.992478e+12 1.657490e+15
## Deg. of Freedom               1        12860
##
## Residual standard error: 359008.6
## Estimated effects may be unbalanced
```

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

```
## I created a data frame that is assigned to a unique variable name known as
uniquehousing which we see the Coefficients for each variable we could enter.
install.packages("outliers")

##
## The downloaded binary packages are in
##  /var/folders/wm/_x82v16s45bgfptwshv23l3r0000gp/T//RtmpNkjeiI/downloaded_p
ackages

library(outliers)

uniquehousing <- lm(formula = sale_price ~ sale_reason + sale_instrument + zi
p_code + building_grade + square_feet_total_living + bedrooms + bath_full_cou
nt + bath_half_count + bath_3qtr_count + year_built + year_renovated + sq_ft_
lot, data = Housing_df)

uniquehousing

##
## Call:
## lm(formula = sale_price ~ sale_reason + sale_instrument + zip_code +
##     building_grade + square_feet_total_living + bedrooms + bath_full_count
+
##     bath_half_count + bath_3qtr_count + year_built + year_renovated +
##     sq_ft_lot, data = Housing_df)
##
## Coefficients:
##              (Intercept)               sale_reason           sale_instrume
nt
```

```
##            -3.931e+07              -1.179e+04                2.430e+
02
##              zip_code          building_grade  square_feet_total_livi
ng
##             3.507e+02               3.077e+04                1.443e+
02
##              bedrooms         bath_full_count         bath_half_cou
nt
##            -3.718e+03              -6.489e+01               -9.841e+
02
##       bath_3qtr_count              year_built           year_renovat
ed
##            -1.568e+04               2.507e+03                7.629e+
01
##             sq_ft_lot
##             2.927e-01
```

```r
## Next I perform an outlier test on the new data frame and the other two exi
sting ones.

outlierTest(uniquehousing)
```

```
##       rstudent unadjusted p-value Bonferroni p
## 11992 10.57364         5.0505e-26   6.4975e-22
## 6430  10.46158         1.6418e-25   2.1122e-21
## 6438  10.44803         1.8918e-25   2.4339e-21
## 6437  10.42057         2.5199e-25   3.2418e-21
## 4649  10.40985         2.8176e-25   3.6248e-21
## 6431  10.32263         6.9642e-25   8.9594e-21
## 6436  10.29156         9.5952e-25   1.2344e-20
## 6441  10.28152         1.0640e-24   1.3689e-20
## 6432  10.25505         1.3969e-24   1.7971e-20
## 6442  10.21695         2.0641e-24   2.6555e-20
```

```r
## Next I will remove outliers as seen below:

Updated_Housing_df <- Housing_df[-c(11992,6430,6438,6437,4649,6431,6436,6441,
6432,6442),]

## Next I perform an outlier test on the square_foot_price_lm

outlierTest(square_foot_price_lm)
```

```
##      rstudent unadjusted p-value Bonferroni p
## 6438 9.334760         1.1763e-20   1.5134e-16
## 6437 9.334494         1.1793e-20   1.5171e-16
## 6441 9.334316         1.1813e-20   1.5197e-16
## 6433 9.334031         1.1844e-20   1.5237e-16
## 6434 9.333823         1.1867e-20   1.5267e-16
## 6430 9.333677         1.1884e-20   1.5288e-16
```

```
## 6442 9.332473         1.2018e-20   1.5462e-16
## 6439 9.331469         1.2132e-20   1.5608e-16
## 6431 9.331388         1.2141e-20   1.5620e-16
## 6429 9.329466         1.2362e-20   1.5904e-16

## Next I will remove outliers as seen below:

Updated_Housing_df <- Housing_df[-c(6438,6437,6441,6433,6434,6430,6442,6439,6
431,6429),]

## Next I perform an outlier test on the sale_price_lm

outlierTest(sale_price_lm)

##       rstudent unadjusted p-value Bonferroni p
## 11992 10.72424         1.0160e-26   1.3070e-22
## 4649  10.59209         4.1537e-26   5.3438e-22
## 6430  10.46710         1.5493e-25   1.9932e-21
## 6437  10.41523         2.6637e-25   3.4269e-21
## 6438  10.40265         3.0366e-25   3.9066e-21
## 6431  10.31155         7.8074e-25   1.0044e-20
## 6436  10.28046         1.0756e-24   1.3838e-20
## 6432  10.17689         3.1066e-24   3.9967e-20
## 6433  10.14913         4.1211e-24   5.3018e-20
## 6434  10.14913         4.1211e-24   5.3018e-20

## Next I will remove outliers as seen below:

Updated_Housing_df <- Housing_df[-c(11992,4649,6430,6437,6438,6431,6436,6432,
6433,6434),]

## I shall show the updated Updated_Housing_df
str(Updated_Housing_df)

## tibble[,24] [12,855 × 24] (S3: tbl_df/tbl/data.frame)
##  $ sale_date              : chr [1:12855] "1/3/06" "1/3/06" "1/3/06" "1/3
/06" ...
##  $ sale_price             : num [1:12855] 698000 649990 572500 420000 369
900 ...
##  $ sale_reason            : num [1:12855] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument        : num [1:12855] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning           : chr [1:12855] "No" "No" "No" "No" ...
##  $ sitetype               : chr [1:12855] "R1" "R1" "R1" "R1" ...
##  $ addr_full              : chr [1:12855] "17021 NE 113TH CT" "11927 178T
H PL NE" "13315 174TH AVE NE" "3303 178TH AVE NE" ...
##  $ zip_code               : num [1:12855] 98052 98052 98052 98052 98052 .
..
##  $ ctyname                : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
REDMOND" ...
##  $ postalctyn             : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
```

```
REDMOND" ...
##  $ lon                      : num [1:12855] -122 -122 -122 -122 -122 ...
##  $ lat                      : num [1:12855] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade           : num [1:12855] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12855] 2810 2880 2770 1620 1440 4160 3
960 3720 4160 2760 ...
##  $ bedrooms                 : num [1:12855] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count          : num [1:12855] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count          : num [1:12855] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count          : num [1:12855] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built               : num [1:12855] 2003 2006 1987 1968 1980 ...
##  $ year_renovated           : num [1:12855] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning           : chr [1:12855] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot                : num [1:12855] 6635 5570 8444 9600 7526 ...
##  $ prop_type                : chr [1:12855] "R" "R" "R" "R" ...
##  $ present_use              : num [1:12855] 2 2 2 2 2 2 2 2 2 2 ...

## updated uniquehousing data frame

updated_square_foot_price_lm <- lm(sale_price ~ sq_ft_lot, data = Updated_Hou
sing_df)

## After that I will updated the lm with all the variables I used

reupdated_square_foot_price_lm <- lm(sale_price ~ sale_reason + sale_instrume
nt + zip_code + building_grade + square_feet_total_living + bedrooms + bath_f
ull_count + bath_half_count + bath_3qtr_count + year_built + year_renovated +
sq_ft_lot, data = Updated_Housing_df)

reupdated_square_foot_price_lm

##
## Call:
## lm(formula = sale_price ~ sale_reason + sale_instrument + zip_code +
##     building_grade + square_feet_total_living + bedrooms + bath_full_count
+
##     bath_half_count + bath_3qtr_count + year_built + year_renovated +
##     sq_ft_lot, data = Updated_Housing_df)
##
## Coefficients:
##              (Intercept)               sale_reason           sale_instrume
nt
##               -1.126e+08                -9.407e+03                -4.161e+
03
##                 zip_code            building_grade  square_feet_total_livi
ng
##                1.104e+03                 3.276e+04                 1.466e+
02
##                 bedrooms           bath_full_count            bath_half_cou
nt
```

```
##              -8.297e+03                    4.313e+03                   -1.863e+
03
##          bath_3qtr_count                   year_built             year_renovat
ed
##              -1.168e+04                    2.208e+03                    7.451e+
01
##                sq_ft_lot
##                1.484e-01
```

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

```
## I decided to update the hatvalues, hatvalues, covariance.ratios, standardi
zed.residuals, studentized.residuals, cooks.distance, and dfbeta.
Updated_Housing_df$dfbeta <- dfbeta(reupdated_square_foot_price_lm)
Updated_Housing_df$leverage <- hatvalues(reupdated_square_foot_price_lm)
Updated_Housing_df$covariance.ratios <- covratio(reupdated_square_foot_price_
lm)
Updated_Housing_df$standardized.residuals <- rstandard(reupdated_square_foot_
price_lm)
Updated_Housing_df$studentized.residuals <- rstudent(reupdated_square_foot_pr
ice_lm)
Updated_Housing_df$cooks.distance <- cooks.distance(reupdated_square_foot_pri
ce_lm)

## I used the str() function to calculate the standardized residuals
str(Updated_Housing_df)
```

```
## tibble[,30] [12,855 × 30] (S3: tbl_df/tbl/data.frame)
##  $ sale_date               : chr [1:12855] "1/3/06" "1/3/06" "1/3/06" "1/3
/06" ...
##  $ sale_price              : num [1:12855] 698000 649990 572500 420000 369
900 ...
##  $ sale_reason             : num [1:12855] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12855] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12855] "No" "No" "No" "No" ...
##  $ sitetype                : chr [1:12855] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12855] "17021 NE 113TH CT" "11927 178T
H PL NE" "13315 174TH AVE NE" "3303 178TH AVE NE" ...
##  $ zip_code                : num [1:12855] 98052 98052 98052 98052 98052 .
..
##  $ ctyname                 : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
REDMOND" ...
##  $ postalctyn              : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
REDMOND" ...
##  $ lon                     : num [1:12855] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12855] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12855] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12855] 2810 2880 2770 1620 1440 4160 3
960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12855] 4 4 4 3 3 4 5 4 4 4 ...
```

```
##  $ bath_full_count         : num [1:12855] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12855] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12855] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12855] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12855] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12855] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12855] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12855] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12855] 2 2 2 2 2 2 2 2 2 2 ...
##  $ dfbeta                  : num [1:12855, 1:13] -88130 -167917 -61105 -25
431 -32783 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:12855] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "(Intercept)" "sale_reason" "sale_instrument" "zip_c
ode" ...
##  $ leverage                : Named num [1:12855] 0.000275 0.00037 0.000382
0.000451 0.000304 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ covariance.ratios       : Named num [1:12855] 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ standardized.residuals  : Named num [1:12855] -0.151 -0.313 -0.256 -0.1
16 -0.166 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ studentized.residuals   : Named num [1:12855] -0.151 -0.312 -0.256 -0.1
16 -0.166 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ cooks.distance          : Named num [1:12855] 4.81e-07 2.78e-06 1.93e-0
6 4.65e-07 6.45e-07 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
```

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

```
## I used large.residual, standardized.residuals > 2, and studentized.residua
ls < -2 to show the sum of large residuals
Updated_Housing_df$large.residual <- Updated_Housing_df$standardized.residual
s > 2 | Updated_Housing_df$studentized.residuals < -2
str(Updated_Housing_df)
```

```
## tibble[,31] [12,855 × 31] (S3: tbl_df/tbl/data.frame)
##  $ sale_date               : chr [1:12855] "1/3/06" "1/3/06" "1/3/06" "1/3
/06" ...
##  $ sale_price              : num [1:12855] 698000 649990 572500 420000 369
900 ...
##  $ sale_reason             : num [1:12855] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12855] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12855] "No" "No" "No" "No" ...
##  $ sitetype                : chr [1:12855] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12855] "17021 NE 113TH CT" "11927 178T
H PL NE" "13315 174TH AVE NE" "3303 178TH AVE NE" ...
##  $ zip_code                : num [1:12855] 98052 98052 98052 98052 98052 .
..
```

```
##  $ ctyname                   : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
REDMOND" ...
##  $ postalctyn                : chr [1:12855] "REDMOND" "REDMOND" "REDMOND" "
REDMOND" ...
##  $ lon                       : num [1:12855] -122 -122 -122 -122 -122 ...
##  $ lat                       : num [1:12855] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade            : num [1:12855] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12855] 2810 2880 2770 1620 1440 4160 3
960 3720 4160 2760 ...
##  $ bedrooms                  : num [1:12855] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count           : num [1:12855] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count           : num [1:12855] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count           : num [1:12855] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built                : num [1:12855] 2003 2006 1987 1968 1980 ...
##  $ year_renovated            : num [1:12855] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning            : chr [1:12855] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot                 : num [1:12855] 6635 5570 8444 9600 7526 ...
##  $ prop_type                 : chr [1:12855] "R" "R" "R" "R" ...
##  $ present_use               : num [1:12855] 2 2 2 2 2 2 2 2 2 2 ...
##  $ dfbeta                    : num [1:12855, 1:13] -88130 -167917 -61105 -25
431 -32783 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:12855] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "(Intercept)" "sale_reason" "sale_instrument" "zip_c
ode" ...
##  $ leverage                  : Named num [1:12855] 0.000275 0.00037 0.000382
0.000451 0.000304 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ covariance.ratios         : Named num [1:12855] 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ standardized.residuals    : Named num [1:12855] -0.151 -0.313 -0.256 -0.1
16 -0.166 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ studentized.residuals     : Named num [1:12855] -0.151 -0.312 -0.256 -0.1
16 -0.166 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ cooks.distance            : Named num [1:12855] 4.81e-07 2.78e-06 1.93e-0
6 4.65e-07 6.45e-07 ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
##  $ large.residual            : Named logi [1:12855] FALSE FALSE FALSE FALSE
FALSE FALSE ...
##   ..- attr(*, "names")= chr [1:12855] "1" "2" "3" "4" ...
```

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

```
## Next we will use the sum to show the sum of the large residuals that evalu
ate as TRUE to be 330
sum(Updated_Housing_df$large.residual)

## [1] 330
```

```
## After we have received our specific variables we can put it into a table.

Updated_Housing_df[Updated_Housing_df$large.residual , c("sale_price", "sale_
reason", "sale_instrument", "zip_code", "building_grade", "square_feet_total_
living", "bedrooms", "bath_full_count", "bath_half_count", "bath_3qtr_count",
"year_built", "year_renovated", "sq_ft_lot")]

## # A tibble: 330 x 13
##    sale_price sale_reason sale_instrument zip_code building_grade
##         <dbl>       <dbl>           <dbl>    <dbl>          <dbl>
##  1     265000           1               3    98053             10
##  2    1390000           1               3    98053              6
##  3     390000           1               3    98052             11
##  4    1588359           1               3    98053              9
##  5    1450000           1               3    98052              8
##  6    1450000           1               3    98052              6
##  7     163000           1               3    98053              9
##  8     270000           1               3    98053             11
##  9     200000           1               3    98053             10
## 10     300000           1               3    98052             11
## # … with 320 more rows, and 8 more variables: square_feet_total_living <db
l>,
## #   bedrooms <dbl>, bath_full_count <dbl>, bath_half_count <dbl>,
## #   bath_3qtr_count <dbl>, year_built <dbl>, year_renovated <dbl>,
## #   sq_ft_lot <dbl>
```

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

```
## Next I use the Updated_Housing_df to calculate the leverage, cooks distanc
e, and covariance rations
Updated_Housing_df[Updated_Housing_df$large.residual , c("leverage" , "cooks.
distance","covariance.ratios")]

## # A tibble: 330 x 3
##    leverage cooks.distance covariance.ratios
##       <dbl>          <dbl>             <dbl>
##  1 0.00135        0.000669             0.996
##  2 0.00296        0.00238              0.993
##  3 0.00142        0.000738             0.996
##  4 0.000787       0.000285             0.997
##  5 0.00101        0.000324             0.998
##  6 0.00233        0.00245              0.990
##  7 0.00119        0.000601             0.996
##  8 0.183          0.186                1.21
##  9 0.00535        0.00497              0.994
## 10 0.000707       0.000278             0.997
## # … with 320 more rows
```

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

```
## I used the durbin Watson test DWT to determine if the necessary calculatio
ns to assess the assumption of independence and state if the condition is met
and I conclude yes it is possible as the Durbin Watson test is about the corr
elation of the residuals. If the data are ordered in some way, you'll get a s
ignificant DW test

dwt(reupdated_square_foot_price_lm)

##  lag Autocorrelation D-W Statistic p-value
##    1       0.7215049     0.5569814       0
##  Alternative hypothesis: rho != 0
```

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

```
## In this calculation we can assess multicollinearity by computing the varia
nce inflation factor known as VIF which measures how much the variance of a r
egression coefficient is inflated due to multicollinearity in the model. As t
he smallest possible value of VIF is one which is the absence of multicolline
arity. While a VIF value that exceeds 5 or 10 indicates a problematic amount
of collinearity.

car::vif(reupdated_square_foot_price_lm)

##              sale_reason          sale_instrument                 zip_code
##                 1.211430                 1.220892                 1.039035
##           building_grade square_feet_total_living                 bedrooms
##                 2.416949                 4.223336                 1.756799
##          bath_full_count          bath_half_count          bath_3qtr_count
##                 2.509261                 1.440674                 2.084983
##               year_built           year_renovated                sq_ft_lot
##                 1.587690                 1.089389                 1.174761

## As seen above the VIF score for the predictor variable square_feet_total_l
iving is the highest we see with a VIF = 4.22 which would be the only one tha
t is close enough to 5 that could even be thought to might be problematic. Bu
t in this case I believe none of the variables in my lm would be problematic.
```

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.

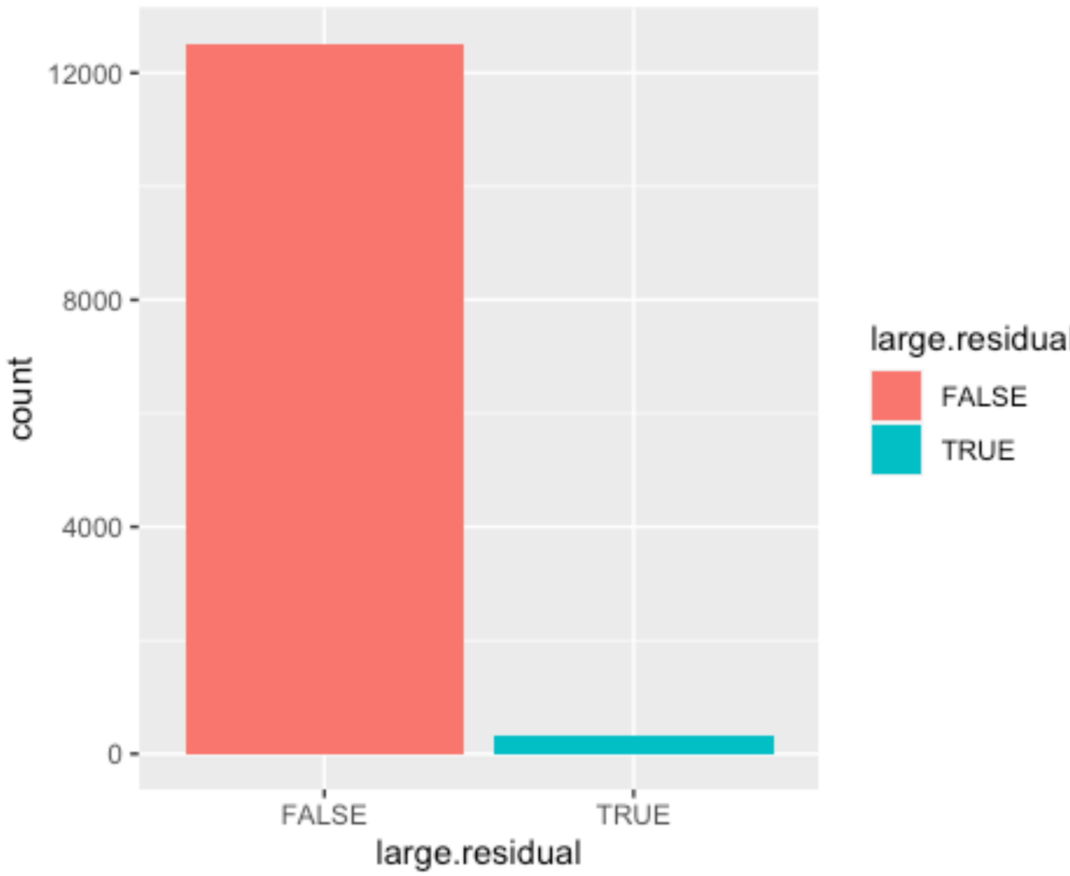
```
## In this graph we see the arg frequency that is related to the residual of
the updated housing data frame
install.packages("Rcmdr")

##
## The downloaded binary packages are in
```

```
##  /var/folders/wm/_x82v16s45bgfptwshv23l3r0000gp/T//RtmpNkjeiI/downloaded_p
ackages

## In this bar chart we saw that out of all our properties we found that over
12,000 are not considered large residual with a small amount being large resi
dual.

ggplot(Updated_Housing_df, aes(large.residual))+geom_bar(aes(fill = large.res
idual))
```




```
with(Updated_Housing_df, Hist(standardized.residuals, scale="frequency", col=
"blue", xlab="Residuals"))

## Error in Hist(standardized.residuals, scale = "frequency", col = "blue", :
could not find function "Hist"
```

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

I believe the regression model is unbiased as we see in our VIF that all our variables are non problematic. With this being said in regards to our sample vs the entire population model I would say it can be biased as we can change our sample to only view variables that we would know that would pair well together as the entire population we would see outliers and data that doesn't really reflect our needs.