# Week 8 - 9

Kimberly Cable

05-14-2022

## Assignment 6

```
##    earn   height    sex ed age  race
## 1 50000 74.42444   male 16  45 white
## 2 60000 65.53754 female 16  58 white
## 3 30000 63.62920 female 16  29 white
## 4 50000 63.10856 female 16  91 other
## 5 51000 63.40248 female 17  39 white
## 6  9000 64.39951 female 15  26 white
```

### Fit a linear model using the `age` variable as the predictor and `earn` as the outcome

lm(outcome ~ predictors, data = dataframe)

```
age_lm <-  lm(earn ~ age, data = heights_df)
```

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

```
summary(age_lm)
```

```
##
## Call:
## lm(formula = earn ~ age, data = heights_df)
##
## 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 ***
## 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
```

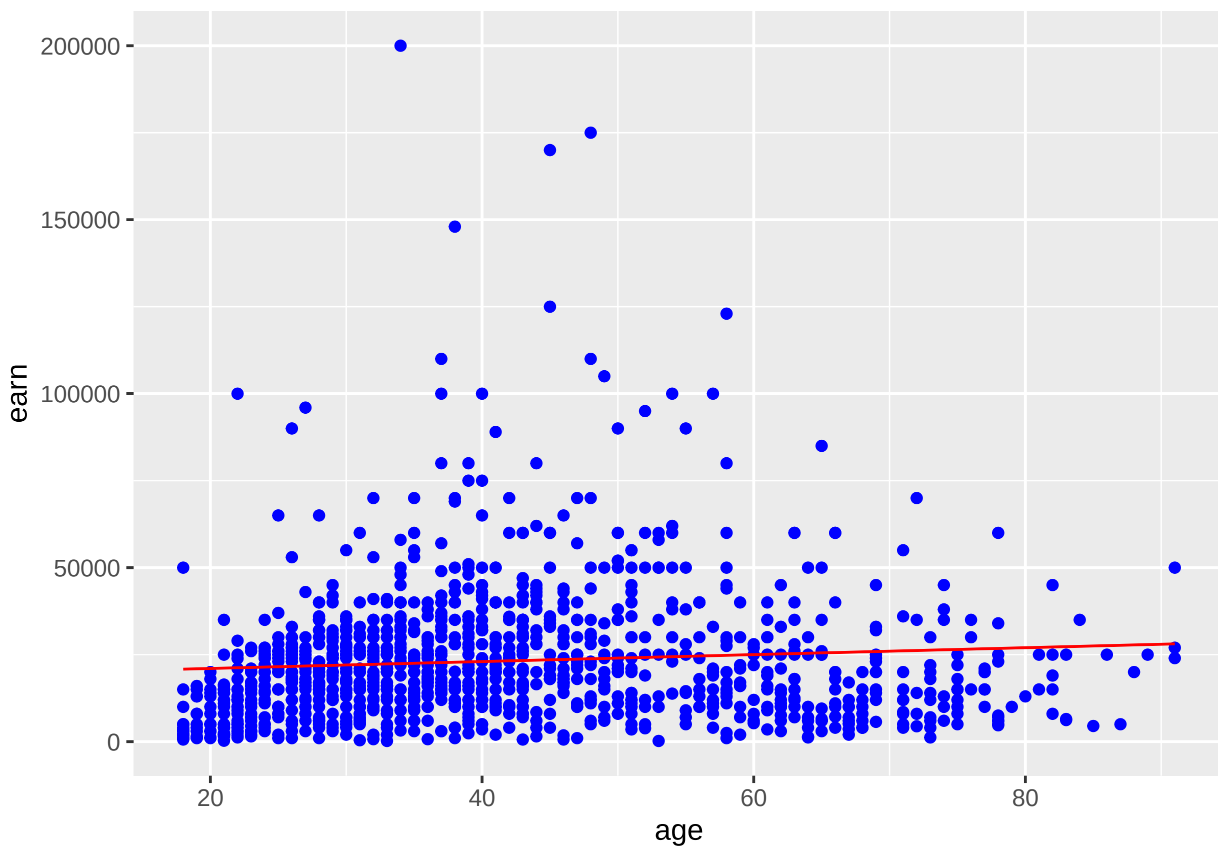
### Creating predictions using predict()

```
age_predict_df <- data.frame(earn = predict(age_lm, heights_df), age = heights_df$age)
head(age_predict_df)
```

```
##       earn age
## 1 23514.79  45
## 2 24807.06  58
## 3 21924.29  29
## 4 28087.45  91
## 5 22918.35  39
## 6 21626.08  26
```

### 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))
```




```
mean_earn <- mean(heights_df$earn)
mean_earn
```

```
## [1] 23154.77
```

### Corrected Sum of Squares Total

```r
sst <- sum((mean_earn - heights_df$earn)^2)
sst
```

```
## [1] 451591883937
```

### Corrected Sum of Squares for Model

```r
ssm <- sum((mean_earn - age_predict_df$earn)^2)
ssm
```

```
## [1] 2963111900
```

### Residuals

```r
residuals <- heights_df$earn - age_predict_df$earn
head(residuals)
```

```
## [1]  26485.214  35192.939   8075.707  21912.549  28081.649 -12626.076
```

### Sum of Squares for Error

```r
sse <- sum(residuals^2)
sse
```

```
## [1] 448628772037
```

### R Squared R^2 = SSM/SST

```r
r_squared <- ssm / sst
r_squared
```

```
## [1] 0.006561482
```

### Number of observations

```r
n <- nrow(heights_df)
n
```

```
## [1] 1192
```

### Number of regression parameters

```r
p <- 2
```

### Corrected Degrees of Freedom for Model (p-1)

```r
dfm <- p - 1
```

### Degrees of Freedom for Error (n-p)

```r
dfe <- n - p
```

### Corrected Degrees of Freedom Total: DFT = n - 1

```r
dft <- n - 1
```

### Mean of Squares for Model: MSM = SSM / DFM

```r
msm <- ssm / dfm
msm
```

```
## [1] 2963111900
```

### Mean of Squares for Error: MSE = SSE / DFE

```r
mse <- sse / dfe
mse
```

```
## [1] 376998968
```

### Mean of Squares Total: MST = SST / DFT

```r
mst <- sst / dft
mst
```

```
## [1] 379170348
```

### F Statistic F = MSM/MSE

```
f_score <- msm / mse
f_score
```

```
## [1] 7.859735
```

### Adjusted R Squared R2 = 1 - (1 - R2)(n - 1) / (n - p)

```
adjusted_r_squared <- 1 - ((1 - r_squared) * dft) / dfe
adjusted_r_squared
```

```
## [1] 0.005726659
```

### Calculate the p-value from the F distribution

```
p_value <- pf(f_score, dfm, dft, lower.tail = F)
p_value
```

```
## [1] 0.005136826
```

## Assignment 7

### Load the `data/r4ds/heights.csv` to

```
##    earn   height    sex ed age  race
## 1 50000 74.42444   male 16  45 white
## 2 60000 65.53754 female 16  58 white
## 3 30000 63.62920 female 16  29 white
## 4 50000 63.10856 female 16  91 other
## 5 51000 63.40248 female 17  39 white
## 6  9000 64.39951 female 15  26 white
```

## Fit a linear model

```
earn_lm <-  lm(earn ~ height + sex + ed + age + race, data = heights_df)
earn_lm
```

```
##
## Call:
## lm(formula = earn ~ height + sex + ed + age + race, data = heights_df)
##
## Coefficients:
##  (Intercept)        height       sexmale            ed           age
##     -41478.5         202.5       10325.6        2768.4         178.3
## racehispanic     raceother     racewhite
##      -1414.3         371.0        2432.5
```

## 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$age,
  race = heights_df$race,
  height = heights_df$height,
  age = heights_df$age,
  sex = heights_df$sex
  )
head(predicted_df)
```

```
##       earn ed  race   height age    sex
## 1 38666.11 45 white 74.42444  45   male
## 2 28859.09 58 white 65.53754  58 female
## 3 23301.90 29 white 63.62920  29 female
## 4 32189.84 91 other 63.10856  91 female
## 5 27807.39 39 white 63.40248  39 female
## 6 20154.60 26 white 64.39951  26 female
```

### Compute deviation (i.e. residuals)

```r
mean_earn <- mean(heights_df$earn)
mean_earn
```

```
## [1] 23154.77
```

### Corrected Sum of Squares Total

```r
sst <- sum((mean_earn - heights_df$earn)^2)
sst
```

```
## [1] 451591883937
```

### Corrected Sum of Squares for Model

```r
ssm <- sum((mean_earn - predicted_df$earn)^2)
ssm
```

```
## [1] 99302918657
```

### Residuals

```r
residuals <- heights_df$earn - predicted_df$earn
head(residuals)
```

```
## [1]  11333.891  31140.911   6698.099  17810.165  23192.610 -11154.599
```

### Sum of Squares for Error

```r
sse <- sum(residuals^2)
sse
```

```
## [1] 3.52289e+11
```

### R Squared

```r
r_squared <- ssm / sst
r_squared
```

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

### Number of observations

```r
n <- nrow(heights_df)
n
```

```
## [1] 1192
```

### Number of regression parameters

```r
p <- 8
```

### Corrected Degrees of Freedom for Model

```r
dfm <- p - 1
dfm
```

```
## [1] 7
```

### Degrees of Freedom for Error

```r
dfe <- n - p
dfe
```

```
## [1] 1184
```

### Corrected Degrees of Freedom Total: DFT = n - 1

```r
dft <- n - 1
dft
```

```
## [1] 1191
```

### Mean of Squares for Model: MSM = SSM / DFM

```r
msm <- ssm / dfm
msm
```

```
## [1] 14186131237
```

### Mean of Squares for Error: MSE = SSE / DFE

```r
mse <- sse / dfe
mse
```

```
## [1] 297541356
```

### Mean of Squares Total: MST = SST / DFT

```r
mst <- sst / dft
mst
```

```
## [1] 379170348
```

### F Statistic

```r
f_score <- msm / mse
f_score
```

```
## [1] 47.67785
```

### Adjusted R Squared R2 = 1 - (1 - R2)(n - 1) / (n - p)

```r
adjusted_r_squared <- 1 - ((1 - r_squared) * dft) / dfe
adjusted_r_squared
```

```
## [1] 0.2152832
```

## Housing Data

**Data for this assignment is focused on real estate transactions recorded from 1964 to 2016 and can be found in Housing.xlsx. Using your skills in statistical correlation, multiple regression, and R programming, you are interested in the following variables: Sale Price and several other possible predictors.**

**If you worked with the Housing dataset in previous week – you are in luck, you likely have already found any issues in the dataset and made the necessary transformations. If not, you will want to take some time looking at the data with all your new skills and identifying if you have any clean up that needs to happen.**

### Load the `data/week-6-housing.csv` to

```
## # A tibble: 4 x 24
##   `Sale Date`         `Sale Price` sale_reason sale_instrument sale_warning
##   <dttm>                     <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00       698000           1               3 <NA>
```

```
## 2 2006-01-03 00:00:00       649990           1                3 <NA>
## 3 2006-01-03 00:00:00       572500           1                3 <NA>
## 4 2006-01-03 00:00:00       420000           1                3 <NA>
## # ... with 19 more variables: sitetype <chr>, addr_full <chr>, zip5 <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>
```

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

I created a variable names total_bath_count that combined bath_full, bath_half, and bath_3qr

I renamed Sale Price to Sale_Price

I renamed Sale Date to Sale_Date

```
## # A tibble: 4 x 25
##   Sale_Date           Sale_Price sale_reason sale_instrument sale_warning
##   <dttm>                   <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00     698000           1               3 <NA>
## 2 2006-01-03 00:00:00     649990           1               3 <NA>
## 3 2006-01-03 00:00:00     572500           1               3 <NA>
## 4 2006-01-03 00:00:00     420000           1               3 <NA>
## # ... with 20 more variables: sitetype <chr>, addr_full <chr>, zip5 <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>, total_bath_count <dbl>
```

#### ii: 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.

For my additional predictors, I chose total_bath_count, square_feet_total_living, and bedrooms because they traditionally are used to calculate the price of a house

```
housing_df_model1 <- lm(Sale_Price ~ sq_ft_lot, data = housing_df)


housing_df_model2 <- lm(Sale_Price ~ total_bath_count + square_feet_total_living +
                            bedrooms, data = housing_df)
```

#### iii: 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?

```
##
```

```
## 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) 641821.40609   3799.91526  168.90 <0.0000000000000002 ***
## sq_ft_lot        0.85099      0.06217   13.69 <0.0000000000000002 ***
## ---
## 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: < 0.00000000000000022
```

The linear regression model for Sales Price and Square Feet per Lot is an R2 = 0.01435 tells us sq feet per lot only accounts for 1.4% of the variation in the Sales Price.

```
##
## Call:
## lm(formula = Sale_Price ~ total_bath_count + square_feet_total_living +
##     bedrooms, data = housing_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1832685  -117901   -40797    44012  3825592
##
## Coefficients:
##                            Estimate Std. Error t value             Pr(>|t|)
## (Intercept)              217573.624  14469.473  15.037 < 0.0000000000000002 ***
## total_bath_count          28490.936   6923.088   4.115       0.000038900092 ***
## square_feet_total_living    185.090      5.078  36.451 < 0.0000000000000002 ***
## bedrooms                 -28001.000   4487.839  -6.239       0.000000000453 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 359500 on 12861 degrees of freedom
## Multiple R-squared:  0.2096, Adjusted R-squared:  0.2095
## F-statistic:  1137 on 3 and 12861 DF,  p-value: < 0.00000000000000022
```

The linear regression model for Sales Price, total_bath_count, square_feet_total_living, and bedrooms is an R2 = 0.2096 tells us that adding other variables accounts for 1.2% of the price of a house.

The adjusted R2 with 1 variable is 0.01428 and with 3 variables it is 0.295 which indicates that adding the additional variables does help but not a lot.

**iii: 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?**

```
##
## 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) 641821.40609   3799.91526  168.90 <0.0000000000000002 ***
## sq_ft_lot        0.85099      0.06217   13.69 <0.0000000000000002 ***
## ---
## 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: < 0.00000000000000022
```

The linear regression model for Sales Price and Square Feet per Lot is an R2 = 0.01435 tells us sq feet per lot only accounts for 1.4% of the variation in the Sales Price.

```
##
## Call:
## lm(formula = Sale_Price ~ total_bath_count + square_feet_total_living +
##     bedrooms, data = housing_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1832685  -117901   -40797    44012  3825592
##
## Coefficients:
##                            Estimate Std. Error t value            Pr(>|t|)
## (Intercept)              217573.624  14469.473  15.037 < 0.0000000000000002 ***
## total_bath_count          28490.936   6923.088   4.115       0.000038900092 ***
## square_feet_total_living    185.090      5.078  36.451 < 0.0000000000000002 ***
## bedrooms                 -28001.000   4487.839  -6.239       0.000000000453 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 359500 on 12861 degrees of freedom
## Multiple R-squared:  0.2096, Adjusted R-squared:  0.2095
## F-statistic:  1137 on 3 and 12861 DF,  p-value: < 0.00000000000000022
```

The linear regression model for Sales Price, total_bath_count, square_feet_total_living, and bedrooms is an R2 = 0.2096 tells us that adding other variables accounts for 1.2% of the price of a house.

The adjusted R2 with 1 variable is 0.01428 and with 3 variables it is 0.295 which indicates that adding the additional variables does help but not a lot.

**iv: 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
beta_model1 <- lm.beta(housing_df_model1)
beta_model1
```

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot, data = housing_df)
##
## Standardized Coefficients::
## (Intercept)   sq_ft_lot
##   0.0000000   0.1198122
```

Standard Deviation of Sale_Price:

```
## [1] 404381.1
```

Standard Deviation of sq_ft_lot:

```
## [1] 56933.29
```

```r
beta_model2 <- lm.beta(housing_df_model2)
beta_model2
```

```
##
## Call:
## lm(formula = Sale_Price ~ total_bath_count + square_feet_total_living +
##     bedrooms, data = housing_df)
##
## Standardized Coefficients::
##              (Intercept)         total_bath_count square_feet_total_living
##               0.00000000               0.04898009               0.45305164
##                 bedrooms
##              -0.06066663
```

Standard Deviation of total_bath_count:

```
## [1] 0.6951902
```

As the number of bath increase by 1 standard deviation (0.70), Sales Price increase by 0.49 standard deviations. So for every 0.7 bathrooms, Sales Price goes up by ($404,381 * 0.049) $19,815.

Standard Deviation of square_feet_total_living:

```
## [1] 989.8176
```

As the number of sq ft of living space increase by 1 standard deviation (990 sq ft), Sales Price increases by 0.453 standard deviations. So for every 990 sq ft added, Sales Price goes up by ($404,381 * 0.453) $183,185.

Standard Deviation of bedrooms:

```
## [1] 0.8761273
```

As the number of bedrooms increases by 1 standard deviation (.88), Sales Price decrease by -0.060 standard deviations. So for every .88 bedrooms added, Sales Price goes down by ($404,381 * -0.06) $24,263.

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

```
model2.confInt <- confint(housing_df_model2)
model2.confInt
```

```
##                                2.5 %      97.5 %
## (Intercept)              189211.3079 245935.9401
## total_bath_count          14920.6568  42061.2160
## square_feet_total_living    175.1371    195.0433
## bedrooms                 -36797.8299 -19204.1695
```

None of the confidence intervals cross 0 but their intervals are not very small so they are less representative in calculating the sales price but are significant

**vi: 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.**

```
## Analysis of Variance Table
##
## Model 1: Sale_Price ~ sq_ft_lot
## Model 2: Sale_Price ~ total_bath_count + square_feet_total_living + bedrooms
##   Res.Df              RSS Df       Sum of Sq      F                Pr(>F)
## 1  12863 2073376756946868
## 2  12861 1662587964360520  2 410788792586347 1588.8 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

```
housing_df$residuals <- resid(housing_df_model2)
housing_df$standarized.residuals <- rstandard(housing_df_model2)
housing_df$studentized.residuals <- rstudent(housing_df_model2)
housing_df$cooks.distance <- cooks.distance(housing_df_model2)
housing_df$dfbeta <- dfbeta(housing_df_model2)
housing_df$dffit <- dffits(housing_df_model2)
housing_df$leverage <- hatvalues(housing_df_model2)
housing_df$covariance.ratios <- covratio(housing_df_model2)

head(housing_df)
```

```
## # A tibble: 6 x 33
##   Sale_Date           Sale_Price sale_reason sale_instrument sale_warning
##   <dttm>                   <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00     698000           1               3 <NA>
## 2 2006-01-03 00:00:00     649990           1               3 <NA>
## 3 2006-01-03 00:00:00     572500           1               3 <NA>
```

```
## 4 2006-01-03 00:00:00     420000           1                3 <NA>
## 5 2006-01-03 00:00:00     369900           1                3 15
## 6 2006-01-03 00:00:00     184667           1               15 18 51
## # ... with 28 more variables: sitetype <chr>, addr_full <chr>, zip5 <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>,
## #   total_bath_count <dbl>, residuals <dbl>, standarized.residuals <dbl>, ...
```

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

```
housing_df$large.residuals <- housing_df$standarized.residuals > 2 |
  housing_df$standarized.residuals < -2
```

**ix: Use the appropriate function to show the sum of large residuals.**

```
sum(housing_df$large.residuals)
```

```
## [1] 322
```

There are 322 cases that have large residuals. So only 2.5% (322/ 12865) of the data was outside the limits.

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

```
housing_df[housing_df$large.residuals, c('Sale_Price', 'total_bath_count',
                                         'square_feet_total_living', 'bedrooms',
                                         'standarized.residuals')]
```

```
## # A tibble: 322 x 5
##    Sale_Price total_bath_count square_feet_total_livi~ bedrooms standarized.res~
##         <dbl>            <dbl>                   <dbl>    <dbl>            <dbl>
##  1     184667             3.25                    4160        4            -2.18
##  2     265000             4.5                     4920        4            -2.45
##  3    1390000             1                        660        0             2.84
##  4     390000             4.5                     5800        5            -2.47
##  5    1588359             2.5                     3360        2             2.04
##  6    1450000             1                        900        2             3.04
##  7     163000             4                       4710        4            -2.58
##  8     270000            23.5                     5060        4            -4.34
##  9     200000             4.5                     6880        5            -3.56
## 10     300000             3.25                    4490        4            -2.03
## # ... with 312 more rows
```

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

```r
large_residuals <- housing_df[housing_df$large.residuals,
                              c('cooks.distance', 'leverage',
                                'covariance.ratios')]
large_residuals
```

```
## # A tibble: 322 x 3
##    cooks.distance leverage covariance.ratios
##             <dbl>    <dbl>             <dbl>
##  1       0.000357 0.000301             0.999
##  2       0.00131  0.000871             0.999
##  3       0.00271  0.00134              0.999
##  4       0.00148  0.000966             0.999
##  5       0.000719 0.000690             1.00
##  6       0.00108  0.000465             0.998
##  7       0.000967 0.000580             0.999
##  8       0.804    0.146                1.16
##  9       0.00525  0.00165              0.998
## 10       0.000440 0.000427             0.999
## # ... with 312 more rows
```

No Cooks distance is > 1 so none of the cases is having an undue influence on the model.

```r
leverage_threshold <- (4 / 12865) * 4

large_residuals['large.leverage'] <-
  large_residuals$leverage > leverage_threshold

sum(large_residuals$large.leverage)
```

```
## [1] 45
```

The average threshold is .0012 so there are only 45 cases greater than the average. With this we seems to have a fairly reliable model that hasnt been unduly influenced by any subset of cases.

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

```r
durbinWatsonTest(housing_df_model2)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7307413     0.5385134       0
##  Alternative hypothesis: rho != 0
```

The durbin watson statistic is 0.539 which is not close to 2 so it has not been met

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

```
vif(housing_df_model2)
```

```
##         total_bath_count square_feet_total_living                 bedrooms
##                 2.305012                 2.513712                 1.538420
```

```
1 / vif(housing_df_model2)
```

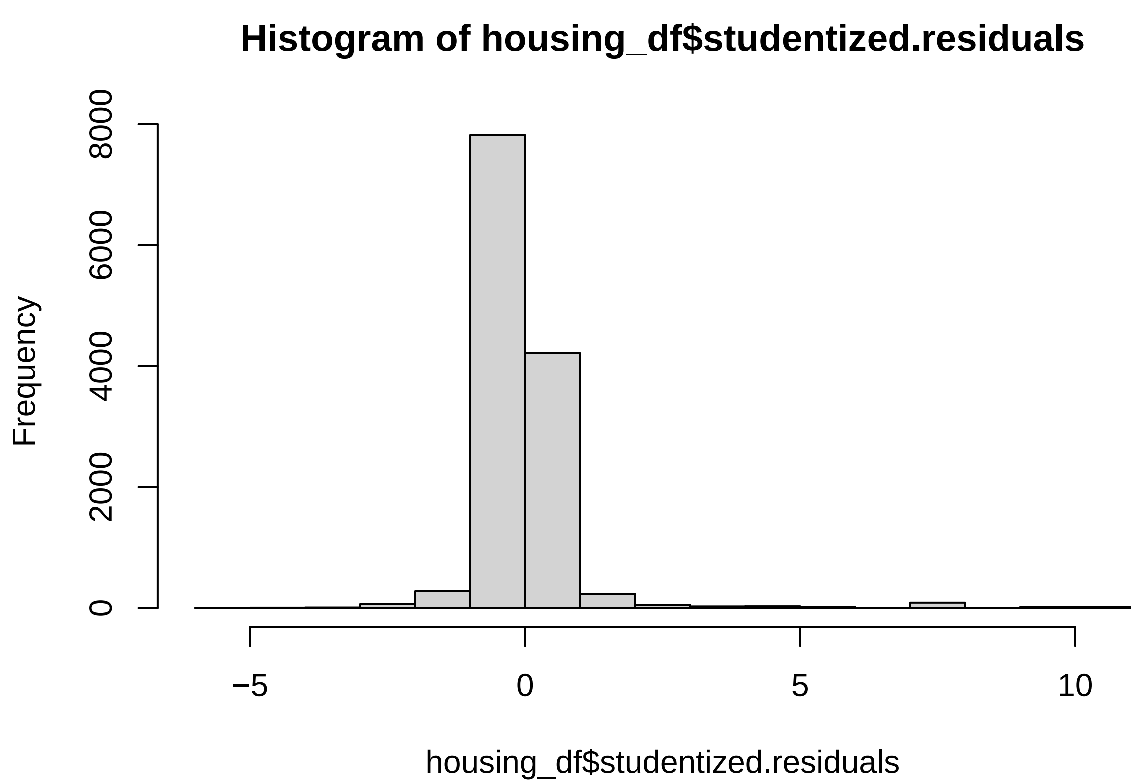
```
##         total_bath_count square_feet_total_living                 bedrooms
##                0.4338373                0.3978181                0.6500174
```

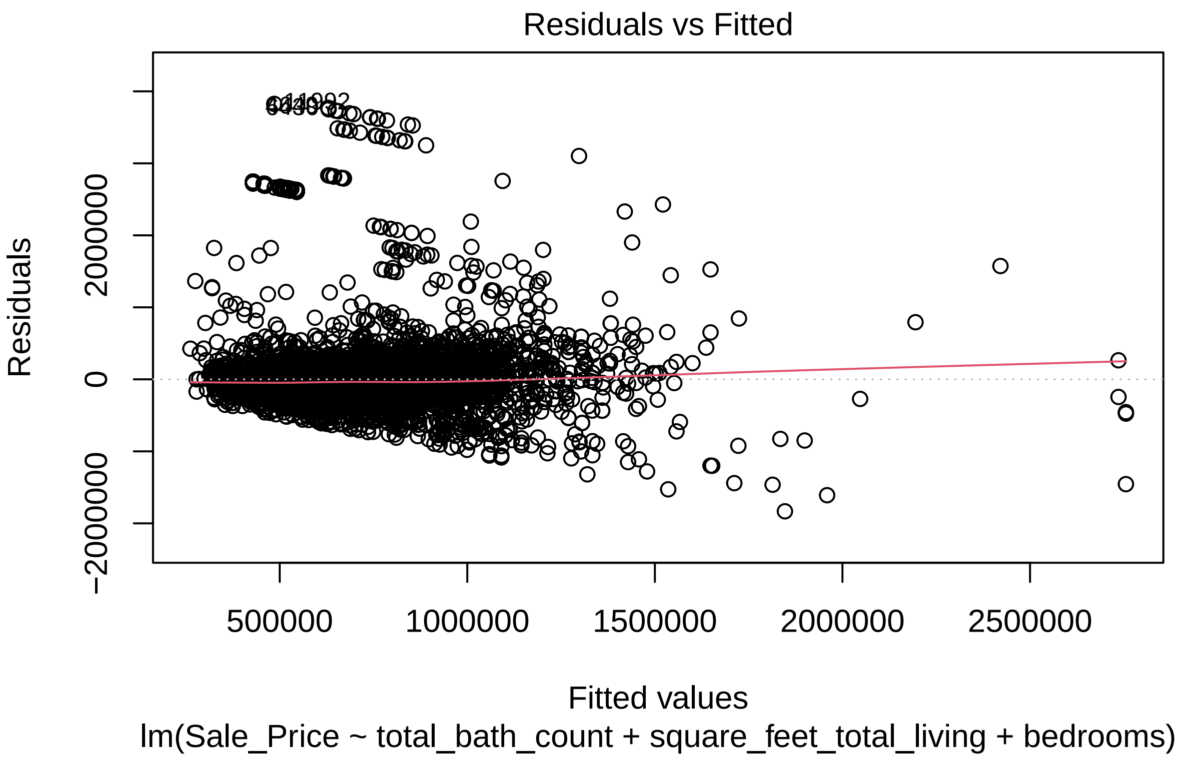
```
mean(vif(housing_df_model2))
```

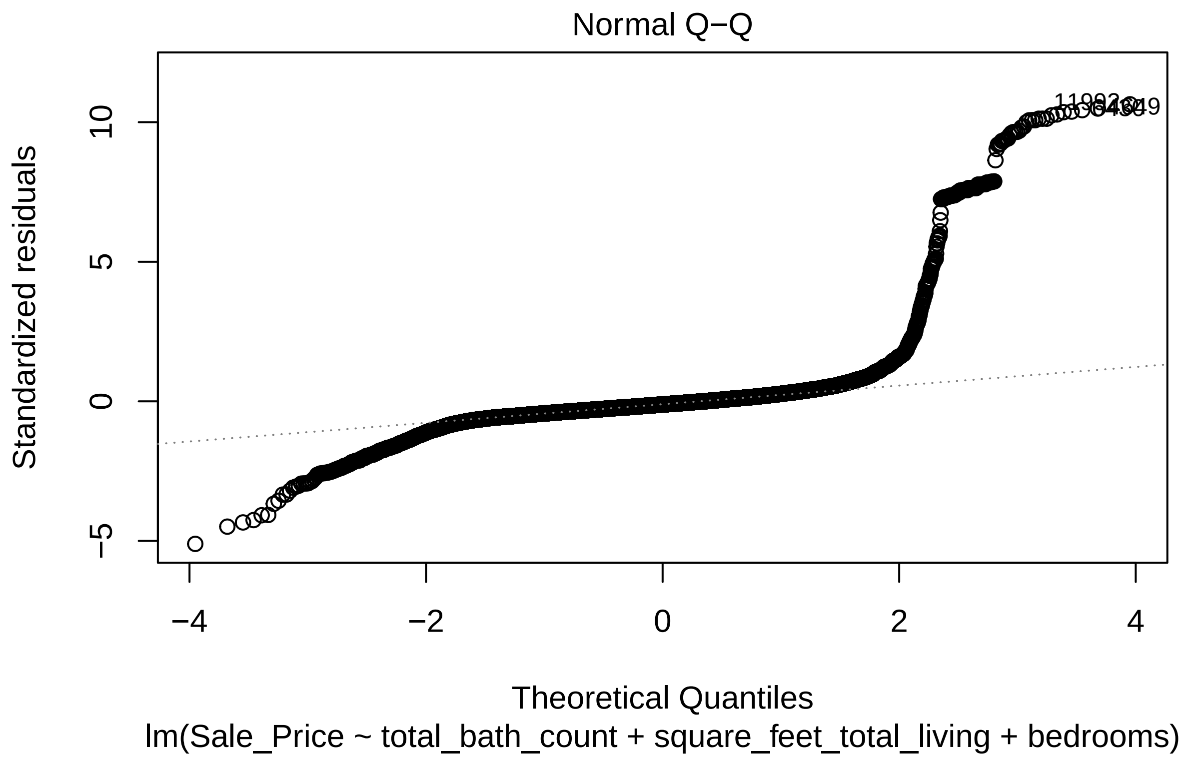
```
## [1] 2.119048
```

The VIF for each variable is below 10, and average VIF for each variable is greater than 1, and the tolerance for all three are above 0.2, and the average VIF is close to 1 so we can conclude that there is no collinearity within the data.

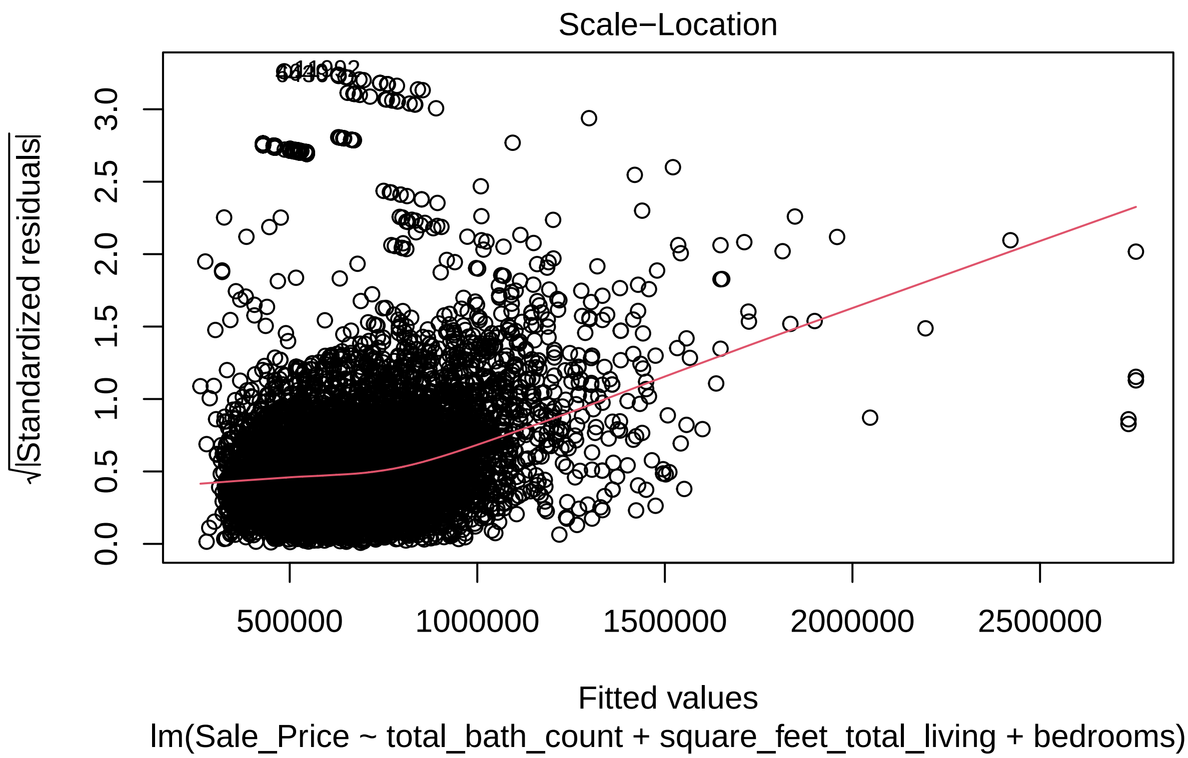
**xiv: Visually check the assumptions related to the residuals using the plot() and hist() functions. Summarize what each graph is informing you of and if any anomalies are present.**

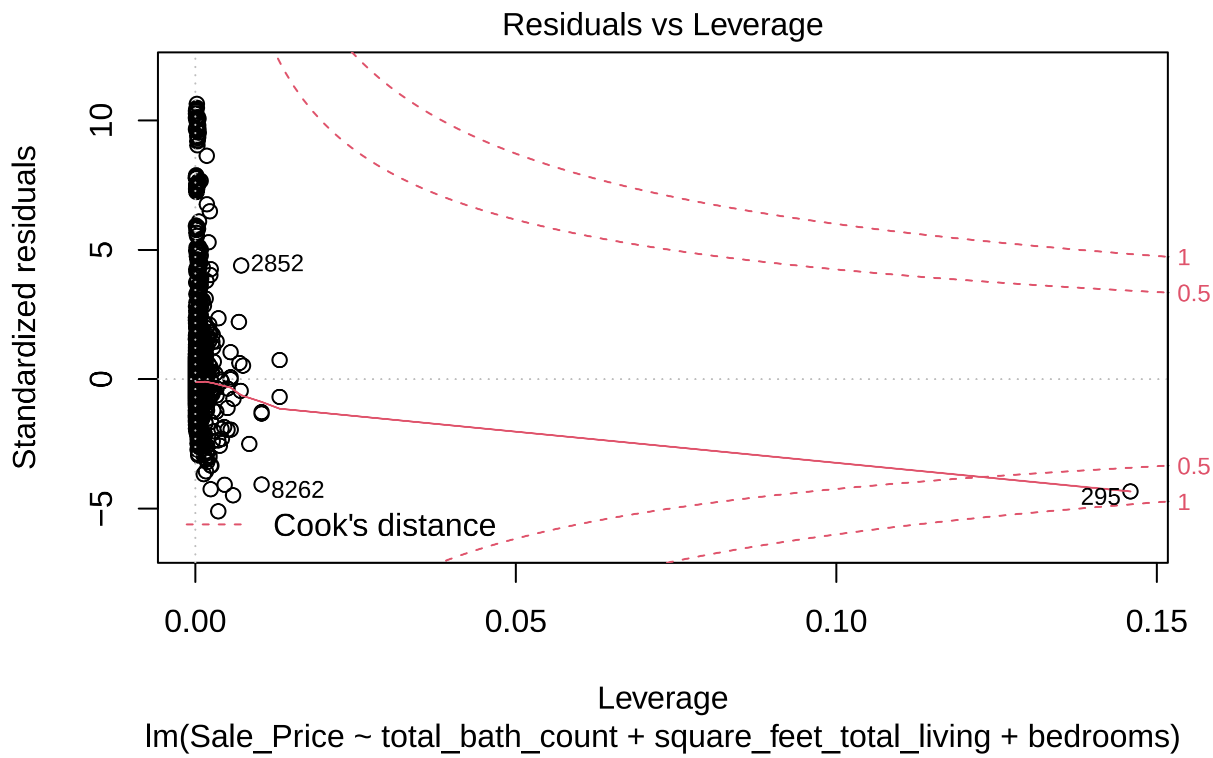



Residuals vs Fitted
Residuals
−2000000
0
2000000
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale_Price ~ total_bath_count + square_feet_total_living + bedrooms)



Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Sale_Price ~ total_bath_count + square_feet_total_living + bedrooms)
−5
0
5
10
−4
−2
0
2
4


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
3.0
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale_Price ~ total_bath_count + square_feet_total_living + bedrooms)




The plots show that is it not normal and skewed

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

The model overall is unbiased and should be ok for the general population.