DSCI445 - Homework 1

Nick Gubbins

Due 9/12/2019 by 4pm

Be sure to `set.seed(400)` at the beginning of your homework.

```
#reproducibility
set.seed(400)
```

R & ggplot2

```
## load the data
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
##
## The following object is masked from 'package:ggplot2':
##
##     last_plot
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(tidymodels)
```

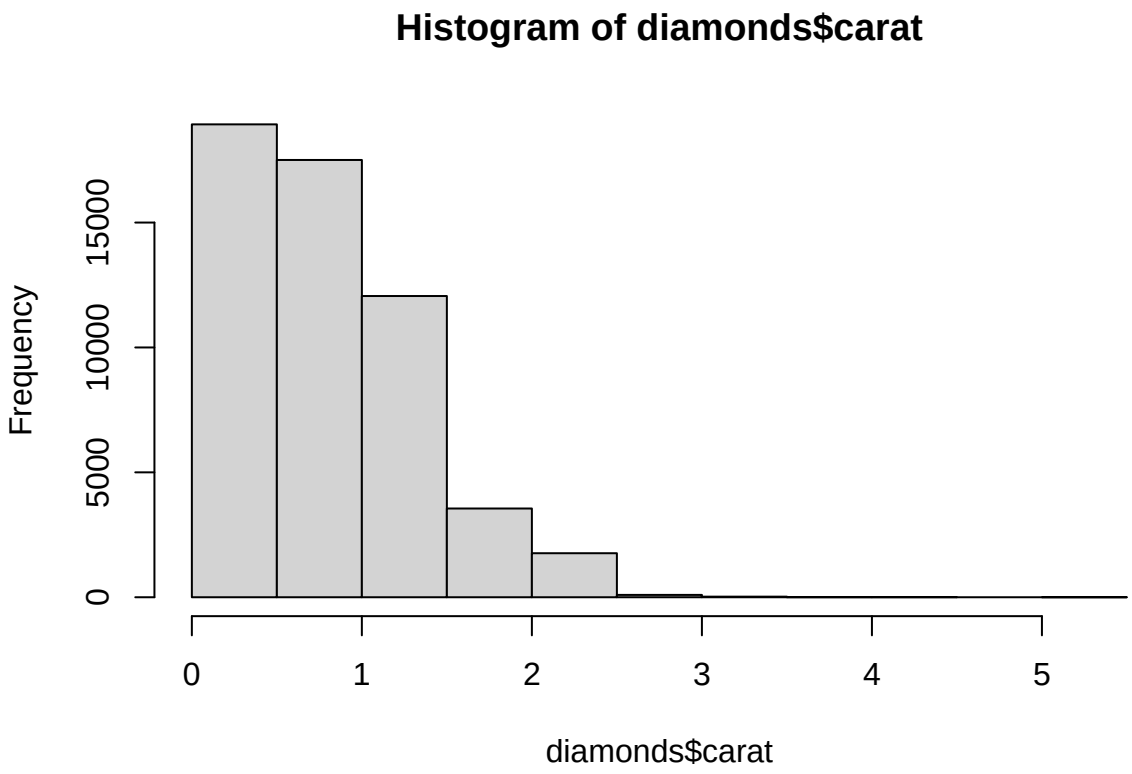
```
## -- Attaching packages --------------------------------------- tidymodels 1.0.0 --
## v broom        1.0.0     v rsample      1.1.0
## v dials        1.0.0     v tune         1.0.0
## v infer        1.0.2     v workflows    1.0.0
```

```
## v modeldata    1.0.0     v workflowsets 1.0.0
## v parsnip      1.0.1     v yardstick    1.0.0
## v recipes      1.0.1
## -- Conflicts ---------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x plotly::filter()  masks dplyr::filter(), stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Use suppressPackageStartupMessages() to eliminate package startup messages
```

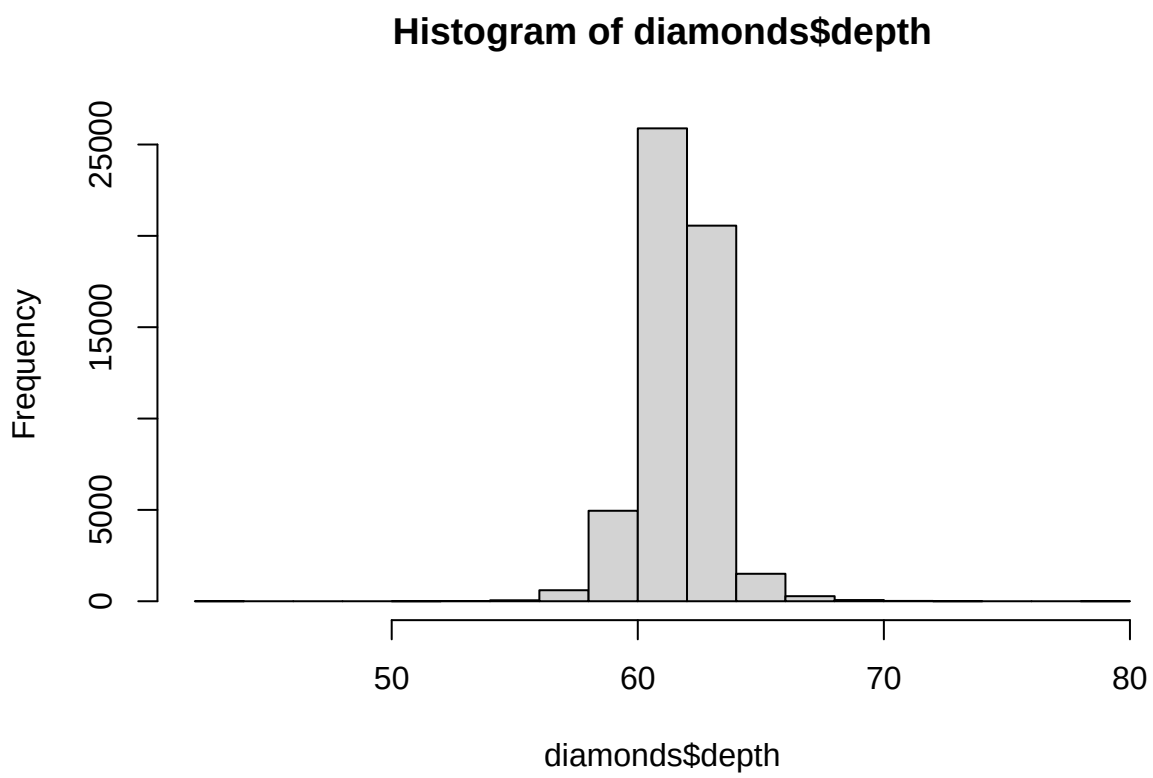
```
## take a look
glimpse(diamonds)
```

```
## Rows: 53,940
## Columns: 10
## $ carat   <dbl> 0.23, 0.21, 0.23, 0.29, 0.31, 0.24, 0.24, 0.26, 0.22, 0.23, 0.~
## $ cut     <ord> Ideal, Premium, Good, Premium, Good, Very Good, Very Good, Ver~
## $ color   <ord> E, E, E, I, J, J, I, H, E, H, J, J, F, J, E, E, I, J, J, J, I,~
## $ clarity <ord> SI2, SI1, VS1, VS2, SI2, VVS2, VVS1, SI1, VS2, VS1, SI1, VS1, ~
## $ depth   <dbl> 61.5, 59.8, 56.9, 62.4, 63.3, 62.8, 62.3, 61.9, 65.1, 59.4, 64~
## $ table   <dbl> 55, 61, 65, 58, 58, 57, 57, 55, 61, 61, 55, 56, 61, 54, 62, 58~
## $ price   <int> 326, 326, 327, 334, 335, 336, 336, 337, 337, 338, 339, 340, 34~
## $ x       <dbl> 3.95, 3.89, 4.05, 4.20, 4.34, 3.94, 3.95, 4.07, 3.87, 4.00, 4.~
## $ y       <dbl> 3.98, 3.84, 4.07, 4.23, 4.35, 3.96, 3.98, 4.11, 3.78, 4.05, 4.~
## $ z       <dbl> 2.43, 2.31, 2.31, 2.63, 2.75, 2.48, 2.47, 2.53, 2.49, 2.39, 2.~
```

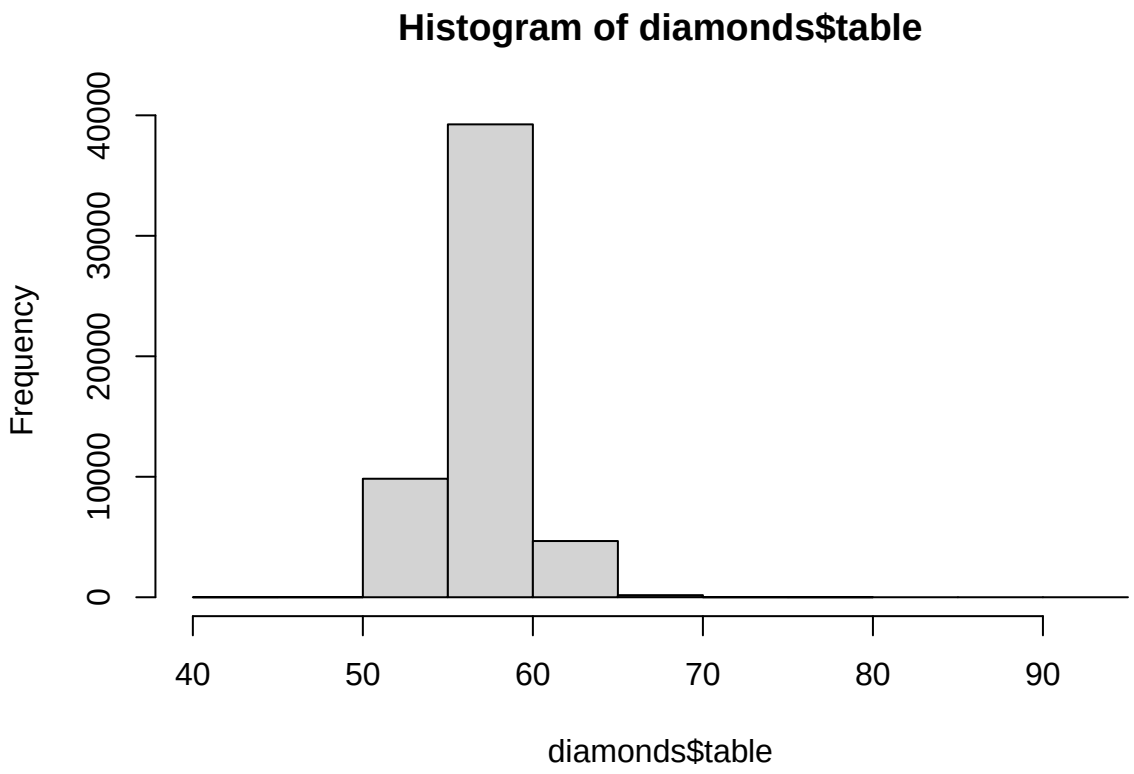
```
## check indiviual variables for issues

hist(diamonds$carat)
```



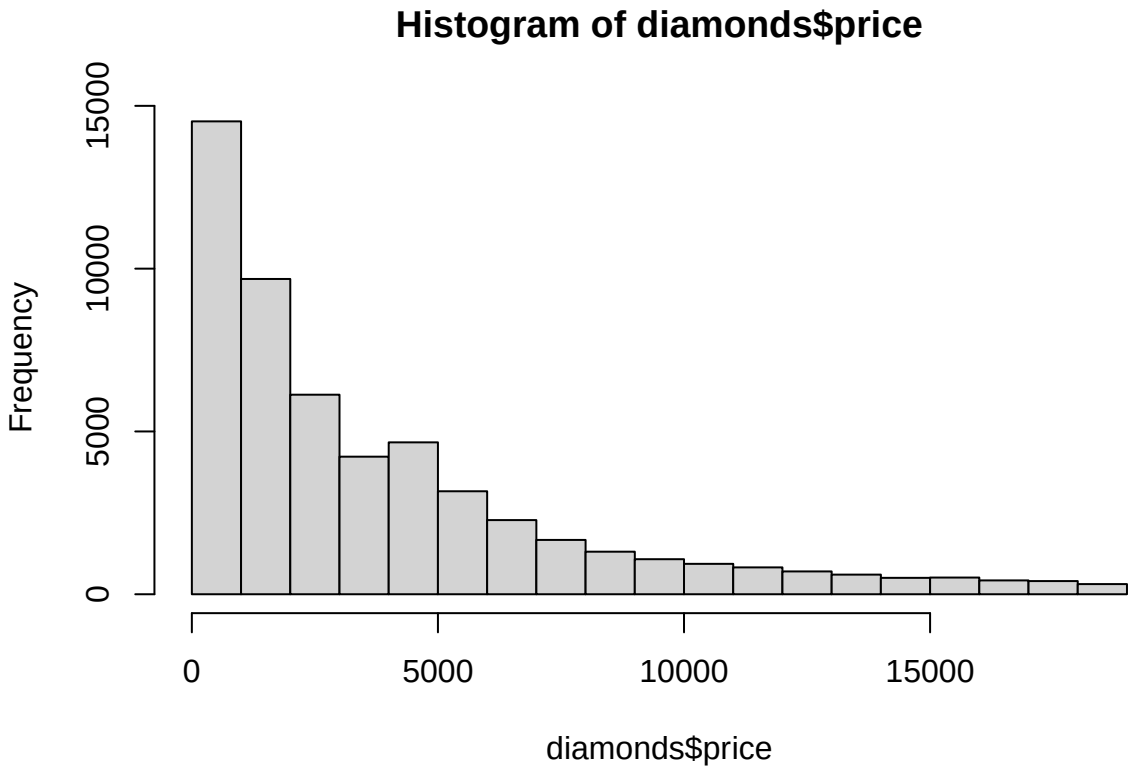

```
hist(diamonds$depth)
```




```
hist(diamonds$table)
```




```
hist(diamonds$price)
```




```
p <- plot_ly(data = diamonds, x=~x, y=~y, z=~z, type="scatter3d", mode="markers")
p
```

Our 3D scatterplot shows that some diamonds only have two dimensions. As we live in a three dimensional world, this is impossible. To improve the data, I will create a new variable called 'Estimated Volume' (aka 'est_vol'). As the diamonds are round cut, a cylindrical estimation of volume is the best I'm willing to attempt for this assignment. I will exclude all impossible diamonds before doing so.

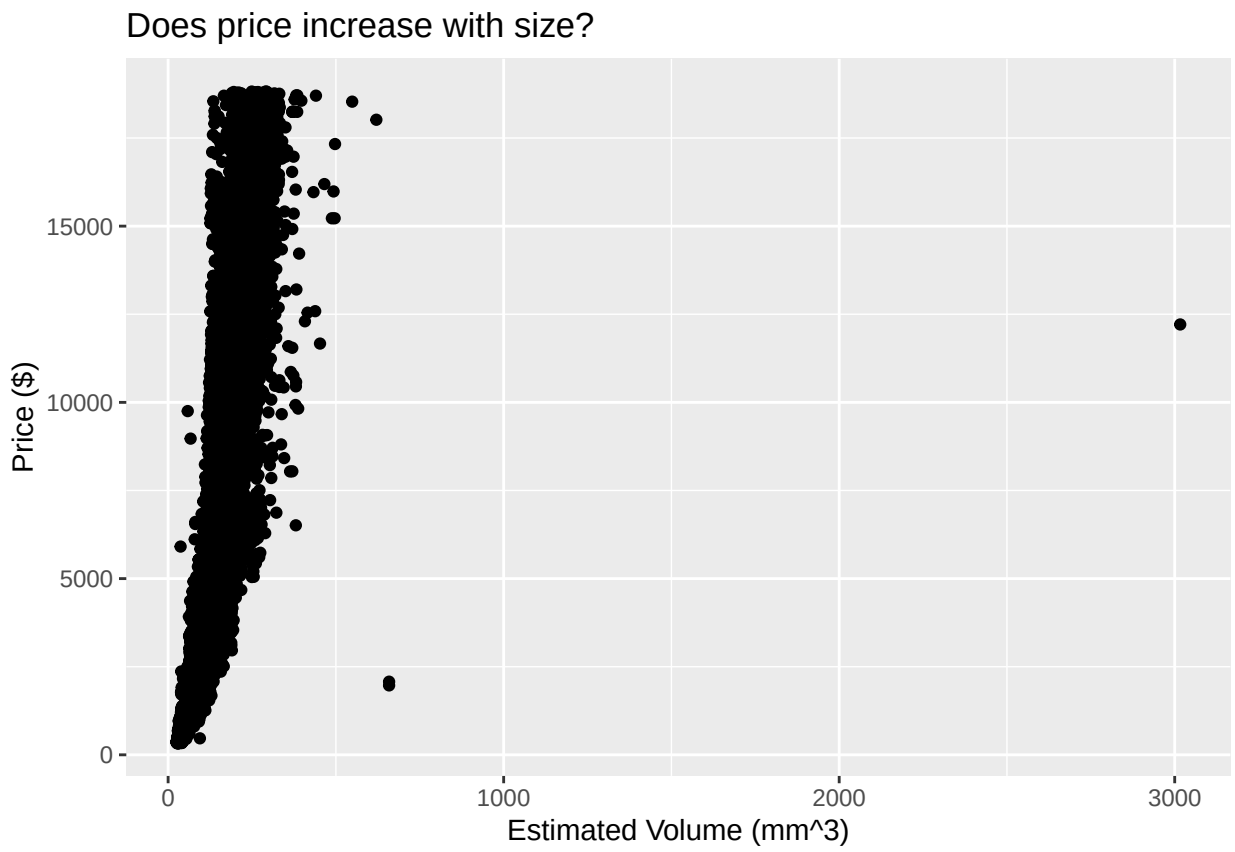
```
new_dia <- diamonds %>%
  filter(x != 0, y != 0, z !=0) %>%
  mutate(est_vol = (3.14159*(x/2)*(y/2)*z) )

glimpse(new_dia)
```

```
## Rows: 53,920
## Columns: 11
## $ carat   <dbl> 0.23, 0.21, 0.23, 0.29, 0.31, 0.24, 0.24, 0.26, 0.22, 0.23, 0.~
## $ cut     <ord> Ideal, Premium, Good, Premium, Good, Very Good, Very Good, Ver~
## $ color   <ord> E, E, E, I, J, J, I, H, E, H, J, J, F, J, E, E, I, J, J, J, I,~
## $ clarity <ord> SI2, SI1, VS1, VS2, SI2, VVS2, VVS1, SI1, VS2, VS1, SI1, VS1, ~
## $ depth   <dbl> 61.5, 59.8, 56.9, 62.4, 63.3, 62.8, 62.3, 61.9, 65.1, 59.4, 64~
## $ table   <dbl> 55, 61, 65, 58, 58, 57, 57, 55, 61, 61, 55, 56, 61, 54, 62, 58~
## $ price   <int> 326, 326, 327, 334, 335, 336, 336, 337, 337, 338, 339, 340, 34~
## $ x       <dbl> 3.95, 3.89, 4.05, 4.20, 4.34, 3.94, 3.95, 4.07, 3.87, 4.00, 4.~
## $ y       <dbl> 3.98, 3.84, 4.07, 4.23, 4.35, 3.96, 3.98, 4.11, 3.78, 4.05, 4.~
## $ z       <dbl> 2.43, 2.31, 2.31, 2.63, 2.75, 2.48, 2.47, 2.53, 2.49, 2.39, 2.~
## $ est_vol <dbl> 30.00378, 27.10081, 29.90549, 36.69737, 40.77568, 30.39013, 30~
```

Now that I have a estimated volumes, I can check to see if price and weight track as I expect they should, with larger diamonds weighing and costing more. I'll have to take the data aggregators word on color, clarity,

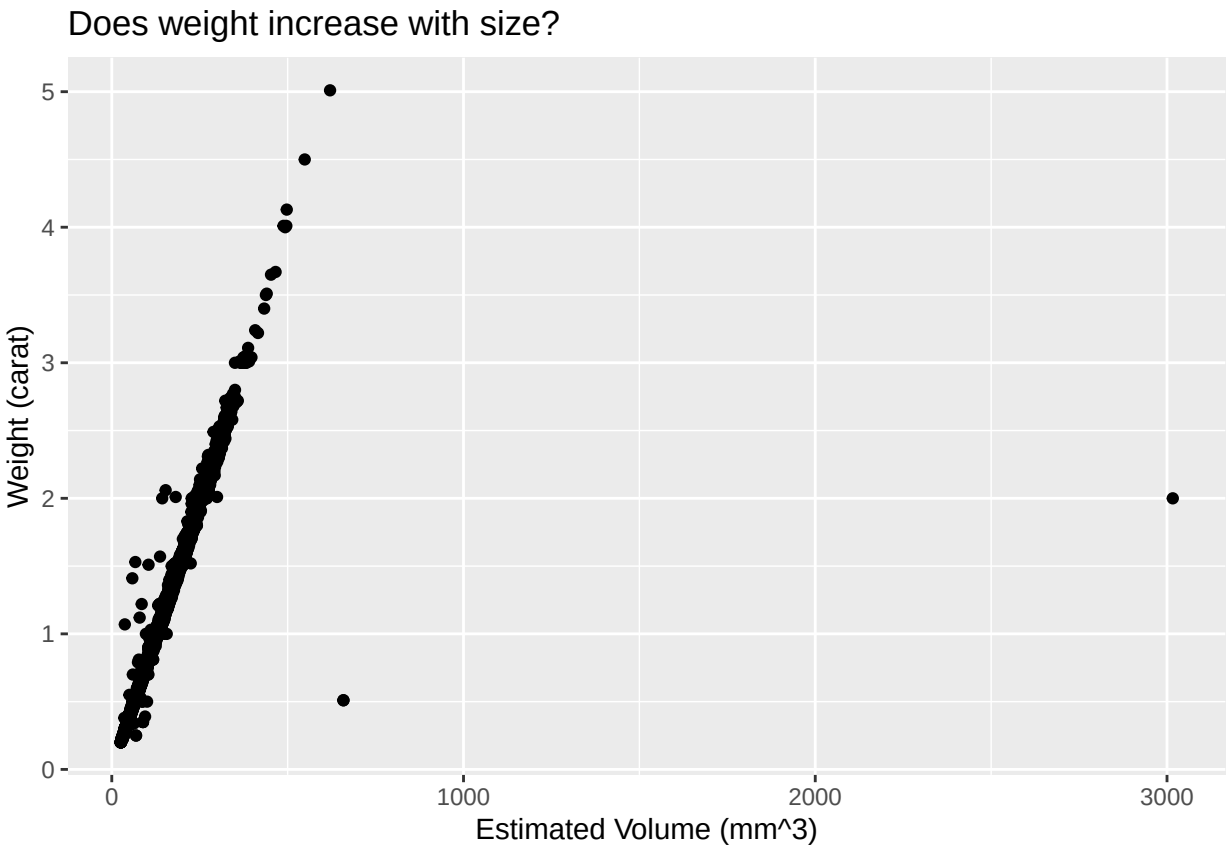
and cut as there isn't a great way to independently check that with the information at hand. While price could be used to validate it, 'for taste there is no accounting'.

```
ggplot(new_dia, aes(x = est_vol, y = price)) +
  geom_point()+
  labs( x = 'Estimated Volume (mm^3)', y = 'Price ($)', title = 'Does price increase with size?')
```



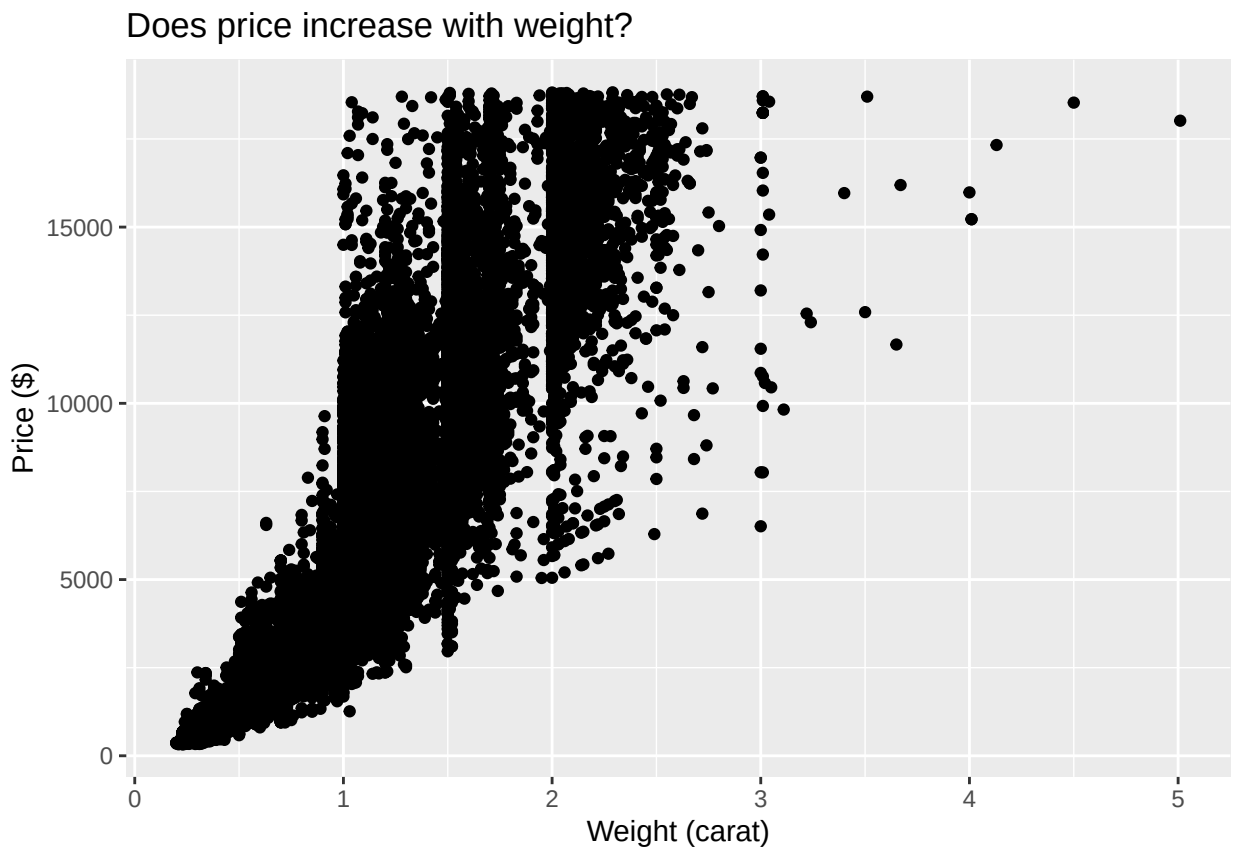
This looks pretty good. There are a few outliers (namely one large diamond that is quite the deal), but on the balance this data looks good. We could check to see if this is valid with

```
ggplot(new_dia, aes(x = est_vol, y = carat)) +
  geom_point()+
  labs( x = 'Estimated Volume (mm^3)', y = 'Weight (carat)', title = 'Does weight increase with size?')
```



This appears to validate the points I previously described as outliers. The 'good deal' diamonds are actually just much less dense than the others. To be assured that this assumption is valid, I need to check if weight and price have a clear relationship.

```
ggplot(new_dia, aes(x = carat, y = price)) +
  geom_point()+
  labs( x = 'Weight (carat)', y = 'Price ($)', title = 'Does price increase with weight?')
```



It appears there is such a relationship. The variability of this relationship could probably be further explained by cut, clarity, and other factors.

Regression

```
## load the data
library(MASS)
```

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

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

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

```
## take a look
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63  2.94
```

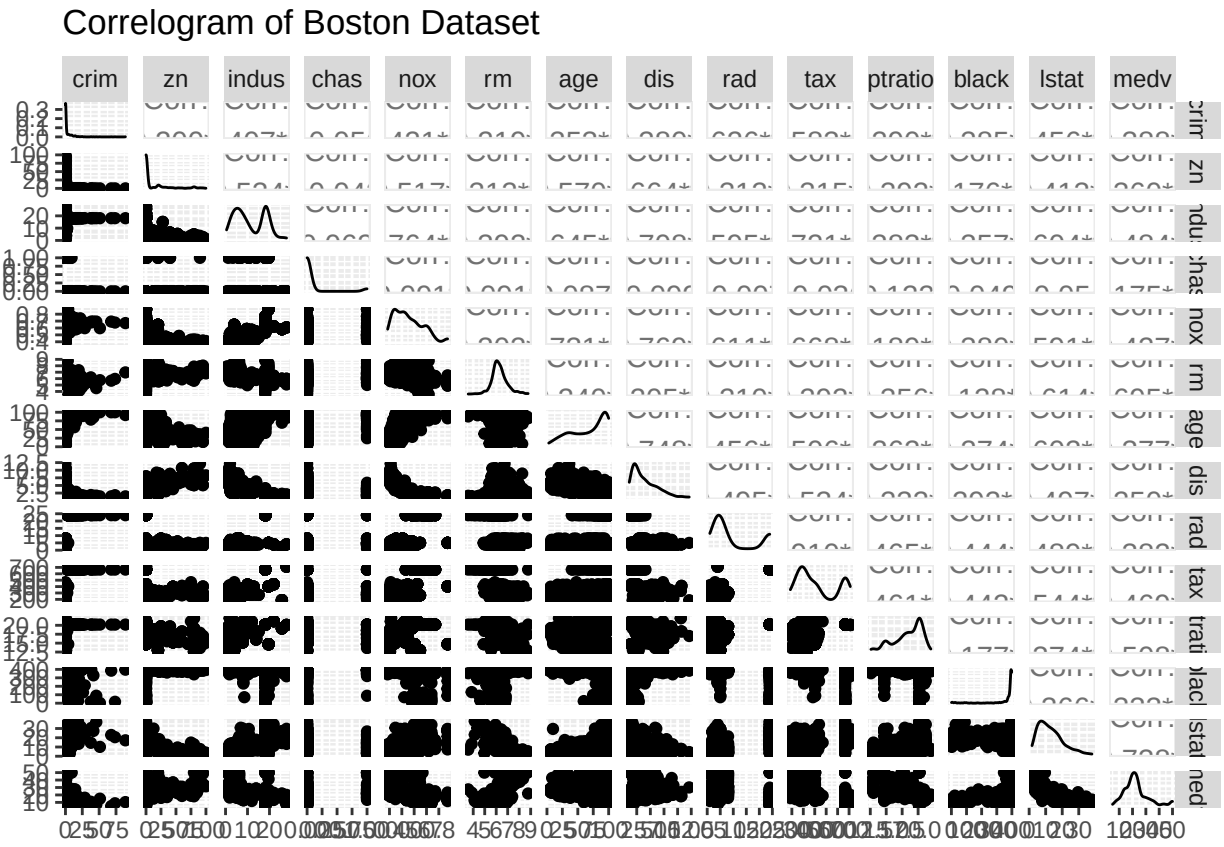
```
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
##   medv
## 1 24.0
## 2 21.6
## 3 34.7
## 4 33.4
## 5 36.2
## 6 28.7
```

Start by visually inspecting the data to get an idea of relationships that might be present (**hint:** look into the `ggpairs` function in the `GGally` package.). Describe what you see.

```
## from the hint
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
p <- ggpairs(data = Boston, title = 'Correlogram of Boston Dataset')

p
```



I found that many of these variables appear to be at least weakly correlated with median home value. The strongest correlation to median home value was with the percentage of lower socioeconomic status people in the area (which may be a product of how the metric calculated), followed by the number of rooms in the house. The weakest correlation with median home value was distance to job centers.

Next fit linear models using the `lm()` function:

(a) For each predictor fit a simple linear regression model to predict the response. Describe your results. In which of the models is there a statistically significant association between the predictor and the response?

```
## # A tibble: 13 x 4
##    var     r_squared  p_value estimate
##    <chr>       <dbl>    <dbl>    <dbl>
##  1 crim       0.149  1.17e-19  -0.415
##  2 zn         0.128  5.71e-17   0.142
##  3 indus      0.232  4.90e-31  -0.648
##  4 chas       0.0288 7.39e- 5   6.35
##  5 nox        0.181  7.07e-24 -33.9
##  6 rm         0.483  2.49e-74   9.10
##  7 age        0.140  1.57e-18  -0.123
##  8 dis        0.0606 1.21e- 8   1.09
##  9 rad        0.144  5.47e-19  -0.403
## 10 tax        0.218  5.64e-29  -0.0256
## 11 ptratio    0.256  1.61e-34  -2.16
## 12 black      0.109  1.32e-14   0.0336
## 13 lstat      0.543  5.08e-88  -0.950
```

I found that all predictors had statistically significant relationships with median home value (with p-values «< 0.05). However, the variance explained by each model tended to be low, ranging from 0.03 to 0.54.

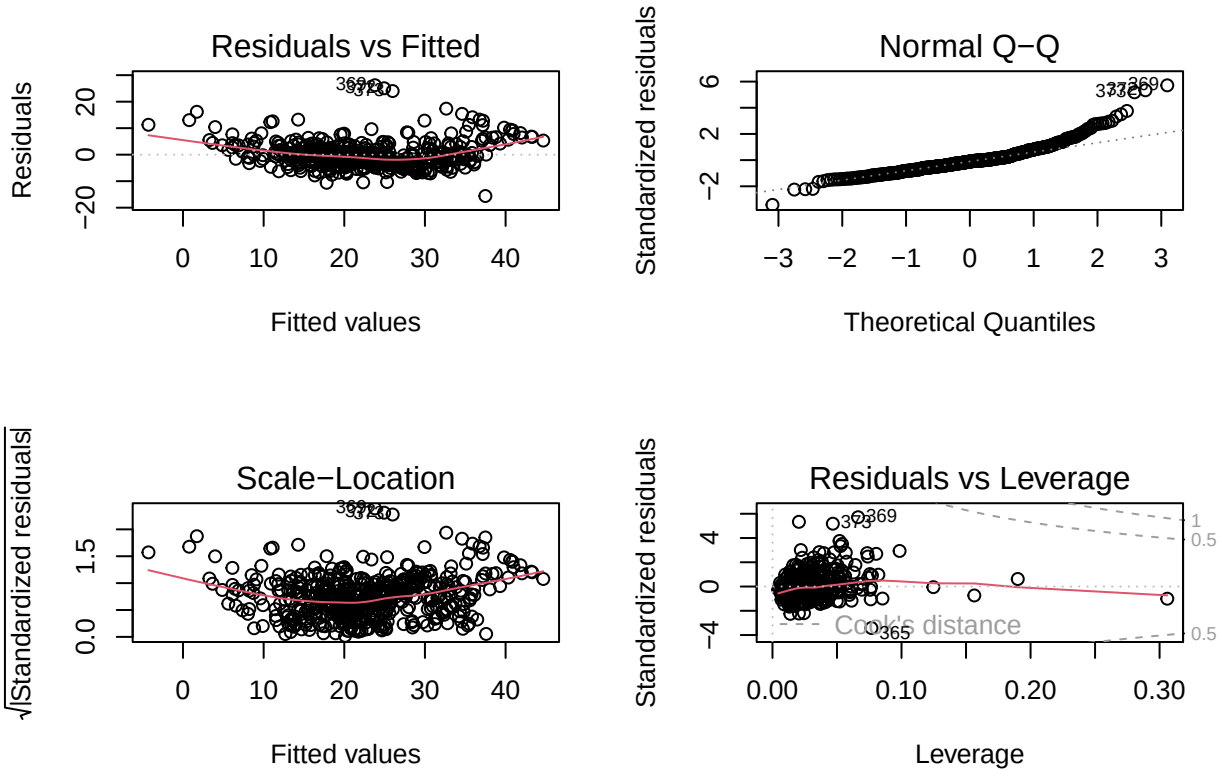
(b) Fit a multiple regression model to predict the response using all of the predictors. Describe your results (including diagnostic plots). For which predictors can we reject the null hypothesis $H_0 : \beta_j = 0$?

```
model <- lm(medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + black + lsta

summary(model)
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +
##     dis + rad + tax + ptratio + black + lstat, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
```

```
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(model)
```



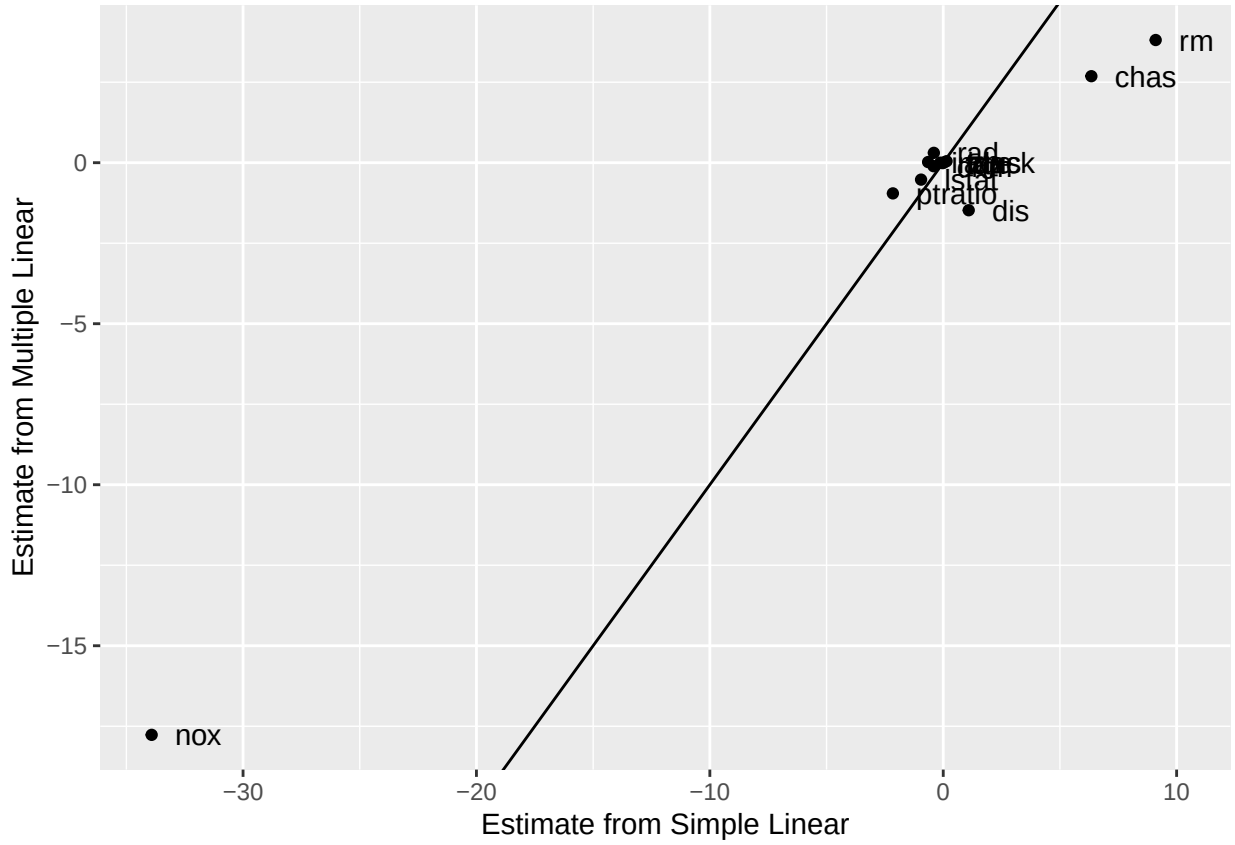
This model appears to be fairly robust for our purposes. The distribution of residuals does not show a clear pattern in our residuals vs fitted plot. Our Normal Q-Q plot shows that, aside from at the highest home values, our model has normally distributed residuals. The relatively flat line in the scale location plot shows that our predictors are fairly homoscedastic. Our residuals vs leverage plot shows no issues with outliers in the model.

We can reject the null hypothesis on all variables except industrial activity in the area and average age. All others show a highly statistically signficiant relationship.

(c) How do your results from (a) compare to your results from (b)? Create a plot displaying the univariate regression coefficients from (a) on the x-axis and the multiple regression coefficients from (b) on the y-axis. That is, each predictor is displayed as a single point on the plot. Its coefficient in a simple linear regression model is shown as its x coordinate and its coefficient in a multiple linear regression model is shown as its y coordinate. Describe what you see.

```
comb_out <- cbind(out[-14,], summary(model)$coefficients[-1,1])
colnames(comb_out)[4] <- 'slr_estimate'
```

```
colnames(comb_out)[5] <- 'mlr_estimate'

ggplot(comb_out, aes(x = slr_estimate, y = mlr_estimate))+
  geom_point()+
  geom_text(aes(label = var), hjust = 0, nudge_x = 1) +
  geom_abline(intercept = 0) +
  labs(x = 'Estimate from Simple Linear',
       y = 'Estimate from Multiple Linear')
```



For most predictors, the estimates produced by both simple and multiple regression were similar. The mulitiple linear regression estimated a much stronger effect from NOx concentrations and the simple linear models estimated a much larger effect from Chas River adjaceny and the number of rooms in each building.

Turn in in a pdf of your analysis to canvas using the provided Rmd file as a template. Your Rmd file on rstudio.cloud will also be used in grading, so be sure they are identical.