BACS-hw11-107070004

Let's go back and take another look at our analysis of the cars dataset. Recall our variables:

1. mpg: miles-per-gallon (dependent variable)
2. cylinders: cylinders in engine
3. displacement: size of engine
4. horsepower: power of engine
5. weight: weight of car
6. acceleration: acceleration ability of car (seconds to achieve 0-60mph)
7. model_year: year model was released
8. origin: place car was designed (1: USA, 2: Europe, 3: Japan)

Did you notice the following from doing a full regression model of mpg on all independent variables?

- Only weight, year, and origin had significant effects
- Non-significant factors cylinders, displacement & horsepower were highly correlated with weight
- Displacement has the opposite effect in the regression from its visualized effect!
- Several factors, like horsepower, seem to have a nonlinear (exponential) relationship with mpg

Question 1) Let's deal with nonlinearity first. Create a new dataset that log-transforms several variables from our original dataset (called cars in this case):

```
cars <- read.table("auto-data.txt", header=FALSE, na.strings = "?")
names(cars) <- c("mpg", "cylinders", "displacement", "horsepower", "weight",
                 "acceleration", "model_year", "origin", "car_name")
cars_log <- with(cars, data.frame(log(mpg), log(cylinders), log(displacement),
                                  log(horsepower), log(weight), log(acceleration),
                                  model_year, origin))
head(cars_log, 5)
```

```
##   log.mpg. log.cylinders. log.displacement. log.horsepower. log.weight.
## 1 2.890372       2.079442          5.726848        4.867534    8.161660
## 2 2.708050       2.079442          5.857933        5.105945    8.214194
## 3 2.890372       2.079442          5.762051        5.010635    8.142063
## 4 2.772589       2.079442          5.717028        5.010635    8.141190
## 5 2.833213       2.079442          5.710427        4.941642    8.145840
##   log.acceleration. model_year origin
## 1          2.484907         70      1
## 2          2.442347         70      1
## 3          2.397895         70      1
## 4          2.484907         70      1
## 5          2.351375         70      1
```

Note: unless you specify column names, each log transformed column is named:.log.

(a) Run a new regression on the cars_log dataset, with log.mpg. dependent on all other variables

```
regr_log <- lm(log.mpg. ~ log.cylinders. + log.displacement. + log.horsepower.
                      + log.weight. + log.acceleration. + model_year
                      + factor(origin), data=cars_log, na.action=na.exclude)
summary(regr_log)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.cylinders. + log.displacement. +
##     log.horsepower. + log.weight. + log.acceleration. + model_year +
##     factor(origin), data = cars_log, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39727 -0.06880  0.00450  0.06356  0.38542
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        7.301938   0.361777  20.184  < 2e-16 ***
## log.cylinders.    -0.081915   0.061116  -1.340  0.18094
## log.displacement.  0.020387   0.058369   0.349  0.72707
## log.horsepower.   -0.284751   0.057945  -4.914 1.32e-06 ***
## log.weight.       -0.592955   0.085165  -6.962 1.46e-11 ***
## log.acceleration. -0.169673   0.059649  -2.845  0.00469 **
## model_year         0.030239   0.001771  17.078  < 2e-16 ***
## factor(origin)2    0.050717   0.020920   2.424  0.01580 *
## factor(origin)3    0.047215   0.020622   2.290  0.02259 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.113 on 383 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.8919, Adjusted R-squared:  0.8897
## F-statistic:   395 on 8 and 383 DF,  p-value: < 2.2e-16
```

(i) Which log-transformed factors have a significant effect on log.mpg. at 10% significance?

ans:

log.horsepower. | ** |
log.weight. | ** |
log.acceleration. | ** |
model_year | *** |
factor(origin)2 | * |
factor(origin)3 | * |

(ii) Do some new factors now have effects on mpg, and why might this be?

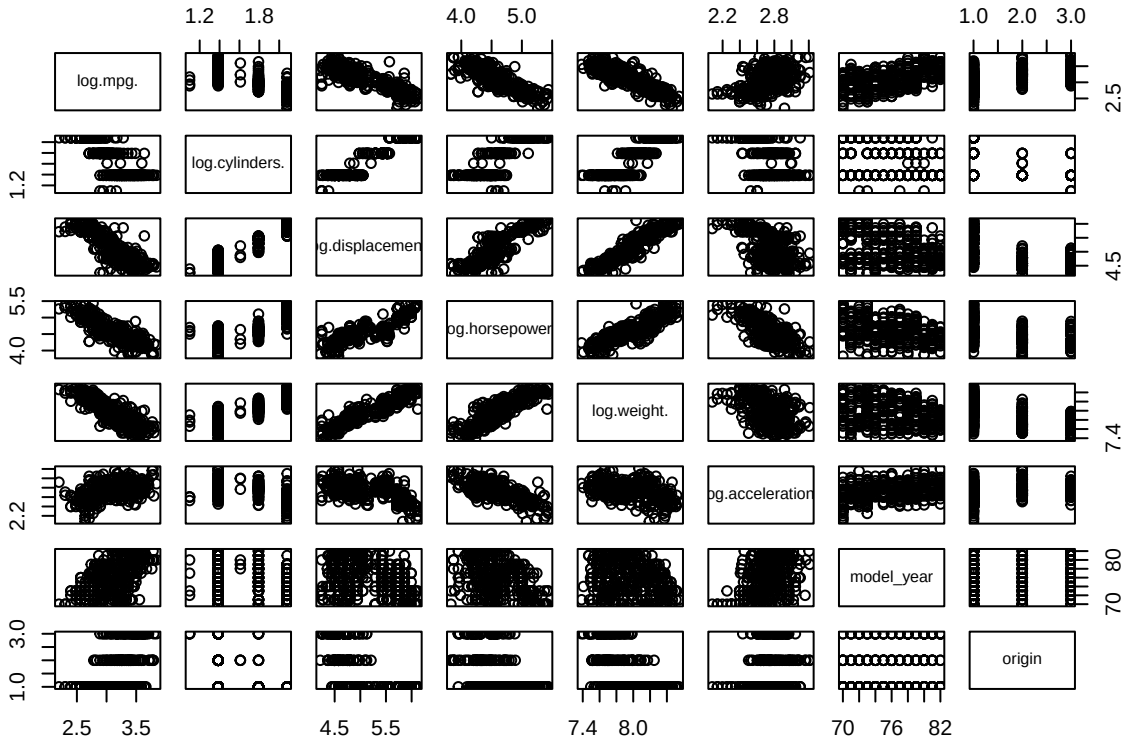
```
cars_m <- data.matrix(cars)
cars_m2 <- as.data.frame(cars_m)
regr <- lm(mpg ~ cylinders + displacement + horsepower+ weight + acceleration
       + model_year + factor(origin), data=cars_m2)
summary(regr)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + model_year + factor(origin), data = cars_m2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0095 -2.0785 -0.0982  1.9856 13.3608
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.795e+01  4.677e+00  -3.839 0.000145 ***
## cylinders       -4.897e-01  3.212e-01  -1.524 0.128215
## displacement     2.398e-02  7.653e-03   3.133 0.001863 **
## horsepower      -1.818e-02  1.371e-02  -1.326 0.185488
## weight          -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
## acceleration     7.910e-02  9.822e-02   0.805 0.421101
## model_year       7.770e-01  5.178e-02  15.005  < 2e-16 ***
## factor(origin)2  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## factor(origin)3  2.853e+00  5.527e-01   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.307 on 383 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

ans: Taking the log of variables will have more symmetric distribution, it will make factors have new effects on mpg. horsepower, accerlation have significant effect on mpg now.

(iii) Which factors still have insignificant or opposite (from correlation) effects on mpg? Why might this be?

```
plot(cars_log)
```



ans:

- insignificant: cylinders
- opposite effects: log.displacement.

(b) Let's take a closer look at weight, because it seems to be a major explanation of mpg

(i) Create a regression (call it regr_wt) of mpg over weight from the original cars dataset

```
regr_wt <- lm(mpg ~ weight, data=cars, na.action=na.exclude)
summary(regr_wt)
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = cars, na.action = na.exclude)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.012  -2.801  -0.351   2.114  16.480
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 46.3173644  0.7952452   58.24   <2e-16 ***
## weight      -0.0076766  0.0002575  -29.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.345 on 396 degrees of freedom
## Multiple R-squared:  0.6918, Adjusted R-squared:  0.691
## F-statistic: 888.9 on 1 and 396 DF,  p-value: < 2.2e-16
```

(ii) Create a regression (call it regr__wt__log) of log.mpg. on log.weight. from cars__log

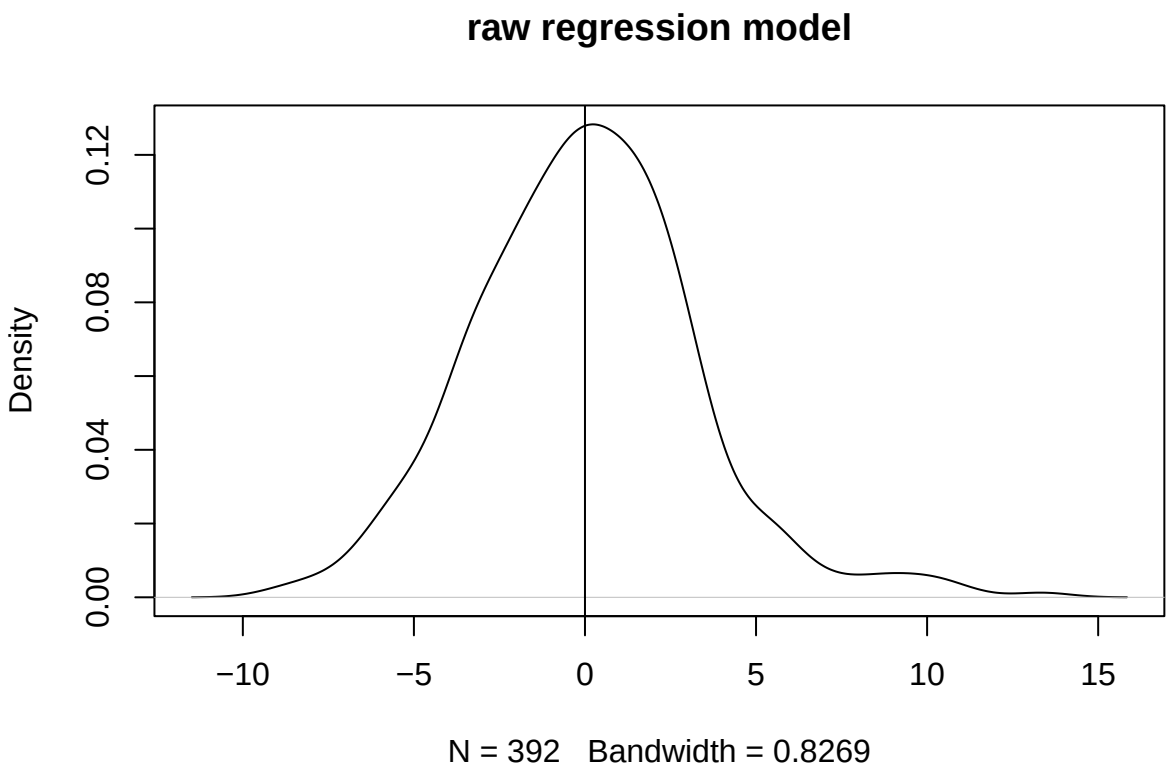
```
regr_wt_log <- lm(log.mpg. ~ log.weight., data=cars_log, na.action=na.exclude)
summary(regr_wt_log)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight., data = cars_log, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52408 -0.10441 -0.00805  0.10165  0.59384
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.5219     0.2349   49.06   <2e-16 ***
## log.weight.  -1.0583     0.0295  -35.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.165 on 396 degrees of freedom
## Multiple R-squared:  0.7647, Adjusted R-squared:  0.7641
## F-statistic:  1287 on 1 and 396 DF,  p-value: < 2.2e-16
```

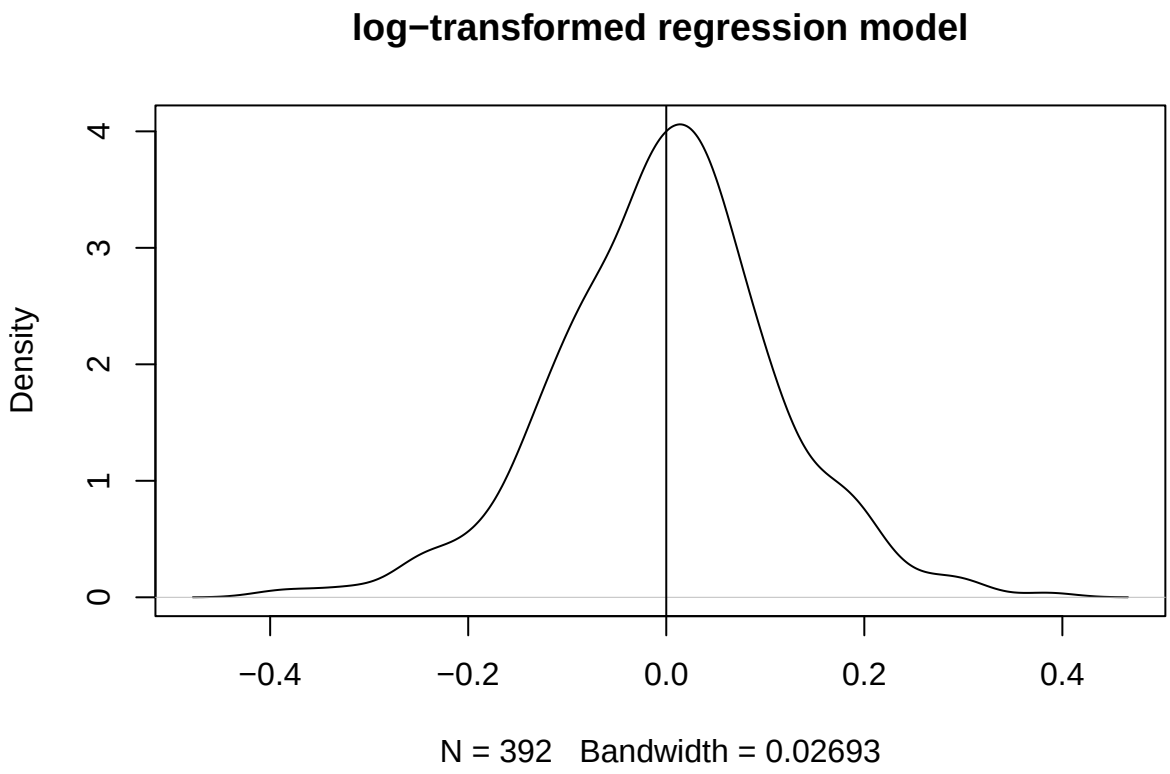
(iii) Visualize the residuals of both regression models (raw and log-transformed):

- density plots of residuals

```
plot(density(regr$residuals), main="raw regression model")
abline(v=mean(regr$residuals))
```

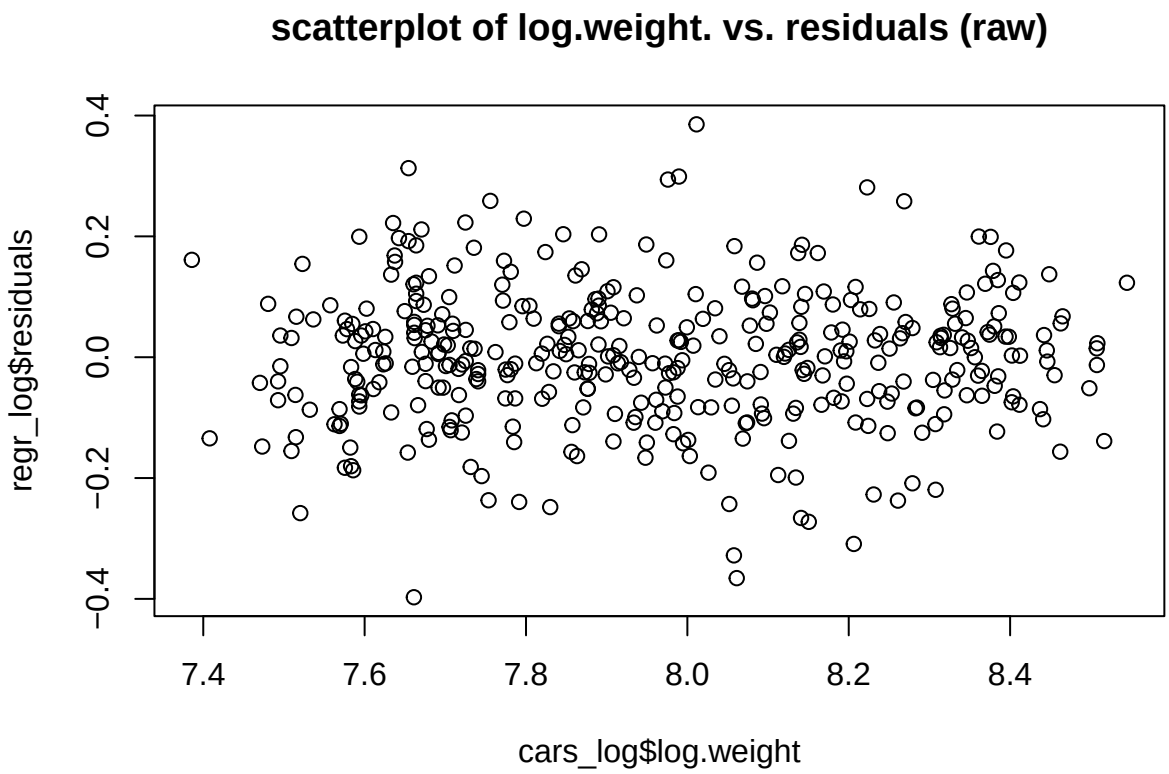



```
plot(density(regr_log$residuals), main="log-transformed regression model")
abline(v=mean(regr_log$residuals))
```

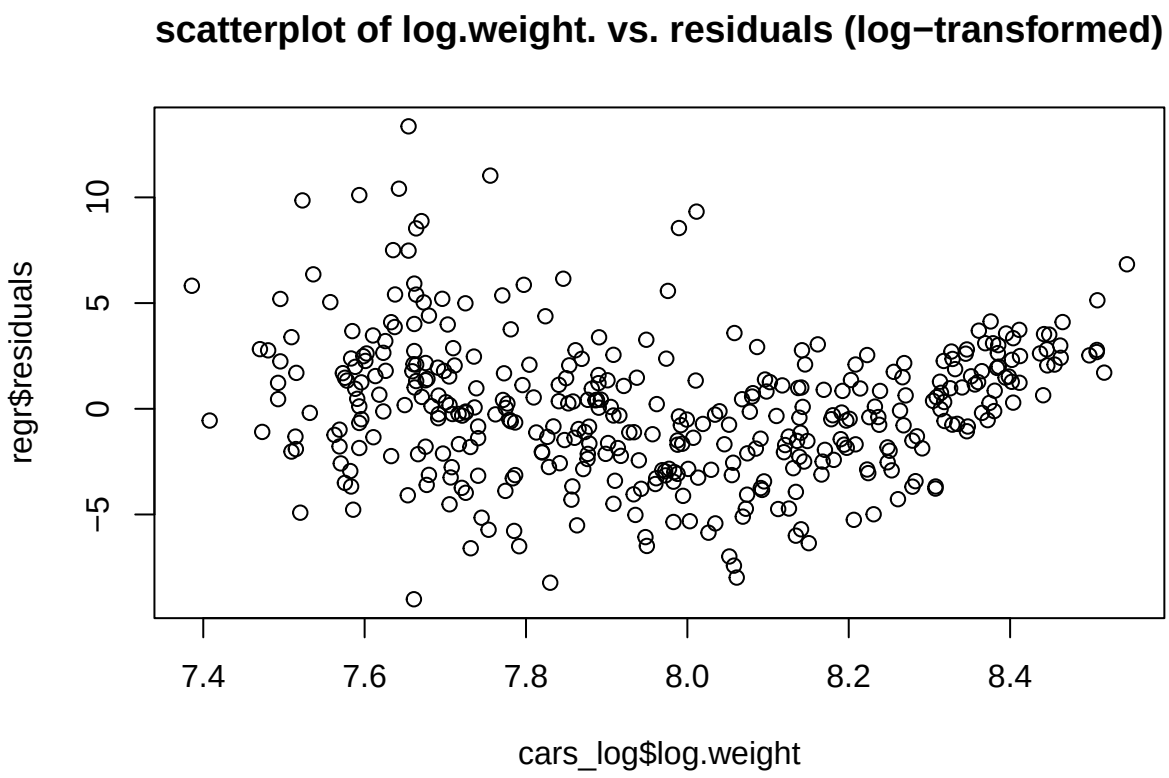


- scatterplot of log.weight. vs. residuals

```
cars_log <- na.omit(cars_log)
plot(cars_log$log.weight, regr_log$residuals
    , main="scatterplot of log.weight. vs. residuals (raw)")
```




```
plot(cars_log$log.weight, regr$residuals
    , main="scatterplot of log.weight. vs. residuals (log-transformed)")
```



(iv) Which regression produces better distributed residuals for the assumptions of regression?

ans: Taking the log of variables will have more symmetric distribution, therefore, log-transformed regression produces better distributed residuals for the assumptions of regression.

(v) How would you interpret the slope of log.weight. vs log.mpg. in simple words?

ans: It is the ratio of the amount that log.mpg. increases as log.weight. increases some amount.

(c) Let's examine the 95% confidence interval of the slope of log.weight. vs. log.mpg.

(i) Create a bootstrapped confidence interval

```
boot_regr <- function(model, dataset) {
  boot_index<-sample(1:nrow(dataset), replace=TRUE)
  data_boot<-dataset[boot_index,]
  regr_boot<-lm(model, data=data_boot)
  regr_boot$coefficients
}
coeffs <- replicate(300, boot_regr(log.mpg.~log.weight., cars_log))
quantile(coeffs["log.weight.",], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## -1.109216 -1.005135
```

(ii) Verify your results with a confidence interval using traditional statistics (i.e., estimate of coefficient and its standard error from lm() results)

```
wt_regr_log <- lm(log.mpg.~log.weight., cars_log)
confint(wt_regr_log)
```

```
##                 2.5 %     97.5 %
## (Intercept) 11.050180 11.9802136
## log.weight. -1.115895 -0.9991175
```

Question 2) Let's tackle multicollinearity next. Consider the regression model:

```
regr_log <- lm(log.mpg. ~ log.cylinders. + log.displacement. + log.horsepower. +
                          log.weight. + log.acceleration. + model_year +
                          factor(origin), data=cars_log)
```

(a) Using regression and R2, compute the VIF of log.weight. using the approach shown in class

```
regr_log_wt <- lm(log.weight. ~ log.cylinders. + log.displacement. + log.horsepower.
                                + log.acceleration. + model_year +
                                factor(origin), data=cars_log)
r2_regr_log_wt <- summary(regr_log_wt)$r.squared
r2_regr_log_wt
```

```
## [1] 0.9431014
```

```
vif_regr_log_wt <- 1/(1-r2_regr_log_wt)
vif_regr_log_wt
```

```
## [1] 17.57512
```

(b) Let's try a procedure called Stepwise VIF Selection to remove highly collinear predictors. Start by Installing the 'car' package in RStudio – it has a function called vif() (note: CAR package stands for Companion to Applied Regression – it isn't about cars!)

(i) Use vif(regr_log) to compute VIF of the all the independent variables

```
vif(regr_log)
```

```
##                        GVIF Df GVIF^(1/(2*Df))
## log.cylinders.    10.456738  1        3.233688
## log.displacement. 29.625732  1        5.442952
## log.horsepower.   12.132057  1        3.483110
## log.weight.       17.575117  1        4.192269
## log.acceleration.  3.570357  1        1.889539
## model_year         1.303738  1        1.141814
## factor(origin)     2.656795  2        1.276702
```

(ii) Eliminate from your model the single independent variable with the largest VIF score that is also greater than 5

1. delete log.displacement.

```
regr_log_d <- lm(log.mpg. ~ log.cylinders.  + log.horsepower. +
                         log.weight. + log.acceleration. + model_year +
                         factor(origin), data=cars_log)
vif(regr_log_d)
```

```
##                        GVIF Df GVIF^(1/(2*Df))
## log.cylinders.     5.433107  1        2.330903
## log.horsepower.   12.114475  1        3.480585
## log.weight.       11.239741  1        3.352572
## log.acceleration.  3.327967  1        1.824272
## model_year         1.291741  1        1.136548
## factor(origin)     1.897608  2        1.173685
```

(iii) Repeat steps (i) and (ii) until no more independent variables have VIF scores above 5

2. delete log.horsepower.

```
regr_log_d <- lm(log.mpg. ~ log.cylinders.  + log.weight.
                         + log.acceleration. + model_year +
                         factor(origin), data=cars_log)
vif(regr_log_d)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.cylinders.    5.427610  1        2.329723
## log.weight.       4.871730  1        2.207200
## log.acceleration. 1.401202  1        1.183724
## model_year        1.206351  1        1.098340
## factor(origin)    1.821167  2        1.161682
```

3. delete log.cylinders.

```
regr_log_d <- lm(log.mpg. ~ log.weight. + log.acceleration.
                         + model_year + factor(origin), data=cars_log)
vif(regr_log_d)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.weight.       1.933208  1        1.390398
## log.acceleration. 1.304761  1        1.142261
## model_year        1.175545  1        1.084225
## factor(origin)    1.710178  2        1.143564
```

There's no more independent variables have VIF scores above 5.

(iv) Report the final regression model and its summary statistics

```
summary(regr_log_d)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight. + log.acceleration. + model_year +
##     factor(origin), data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38259 -0.07054  0.00401  0.06696  0.39798
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        7.410974   0.316806  23.393  < 2e-16 ***
## log.weight.       -0.875499   0.029086 -30.101  < 2e-16 ***
## log.acceleration.  0.054377   0.037132   1.464  0.14389
## model_year         0.032787   0.001731  18.937  < 2e-16 ***
## factor(origin)2    0.056111   0.018241   3.076  0.00225 **
## factor(origin)3    0.031937   0.018506   1.726  0.08519 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1163 on 386 degrees of freedom
## Multiple R-squared:  0.8845, Adjusted R-squared:  0.883
## F-statistic: 591.1 on 5 and 386 DF,  p-value: < 2.2e-16
```

(c) Using stepwise VIF selection, have we lost any variables that were previously significant? If so, how much did we hurt our explanation by dropping those variables? (hint: look at model fit)

- ans 1: We lose log.horsepower. that was previously significant.
- ans 2: the R-square is from 0.8919 to 0.8845, therefore the model fit is less than the original one, it hurt a little but okay.

(d) From only the formula for VIF, try deducing/deriving the following:

(i) If an independent variable has no correlation with other independent variables, what would its VIF score be?

From the formula we can know that the vif=1/(1-R^2), therefore when an independent variable has no correlation with other independent variables, the R=0. We know that vif=1/(1-0^2)~=1.

(ii) Given a regression with only two independent variables (X1 and X2), how correlated would X1 and X2 have to be, to get VIF scores of 5 or higher? To get VIF scores of 10 or higher?

If vif=1/(1-R^2)>=5, the correlation of X1 and X2 have to be

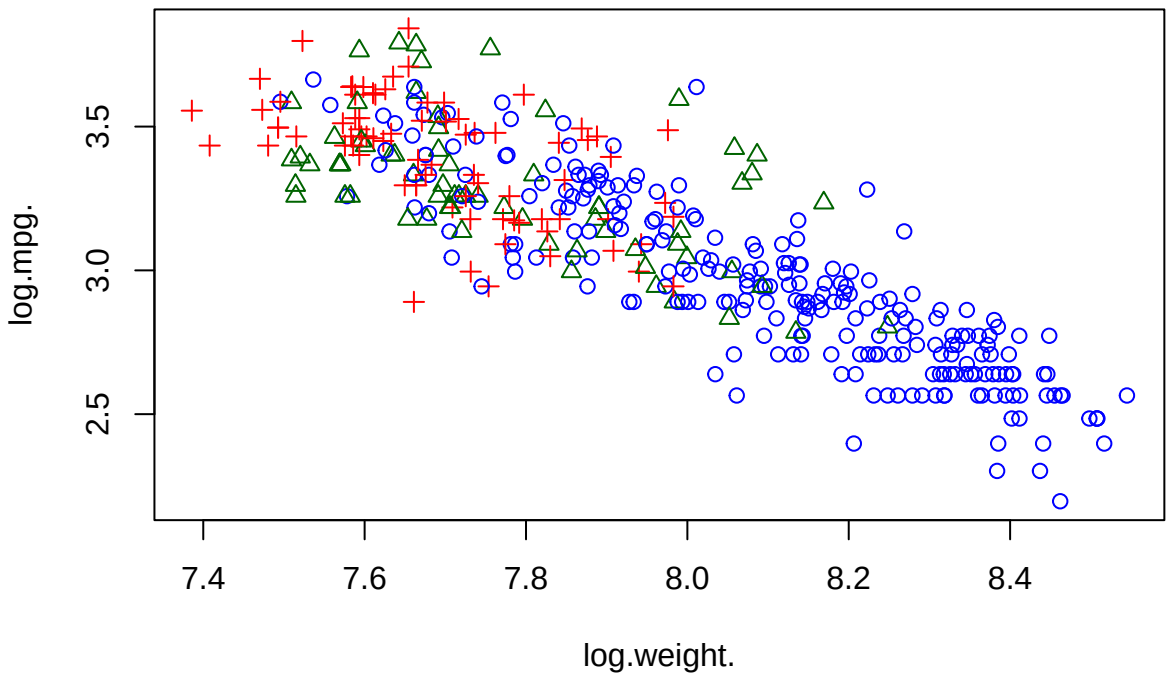
$$(-1, \frac{-2\sqrt{5})}{5}]U[\frac{2\sqrt{5}}{5}, 1)$$

If vif=1/(1-R^2)>=10, the correlation of X1 and X2 have to be

$$(-1, \frac{-3\sqrt{10}}{10}]U[\frac{3\sqrt{10}}{10}, 1)$$

Question 3) Might the relationship of weight on mpg be different for cars from different origins? Let's try visualizing this. First, plot all the weights, using different colors and symbols for the three origins:

```
origin_colors = c("blue", "darkgreen", "red")
with(cars_log, plot(log.weight., log.mpg., pch=origin, col=origin_colors[origin]))
```



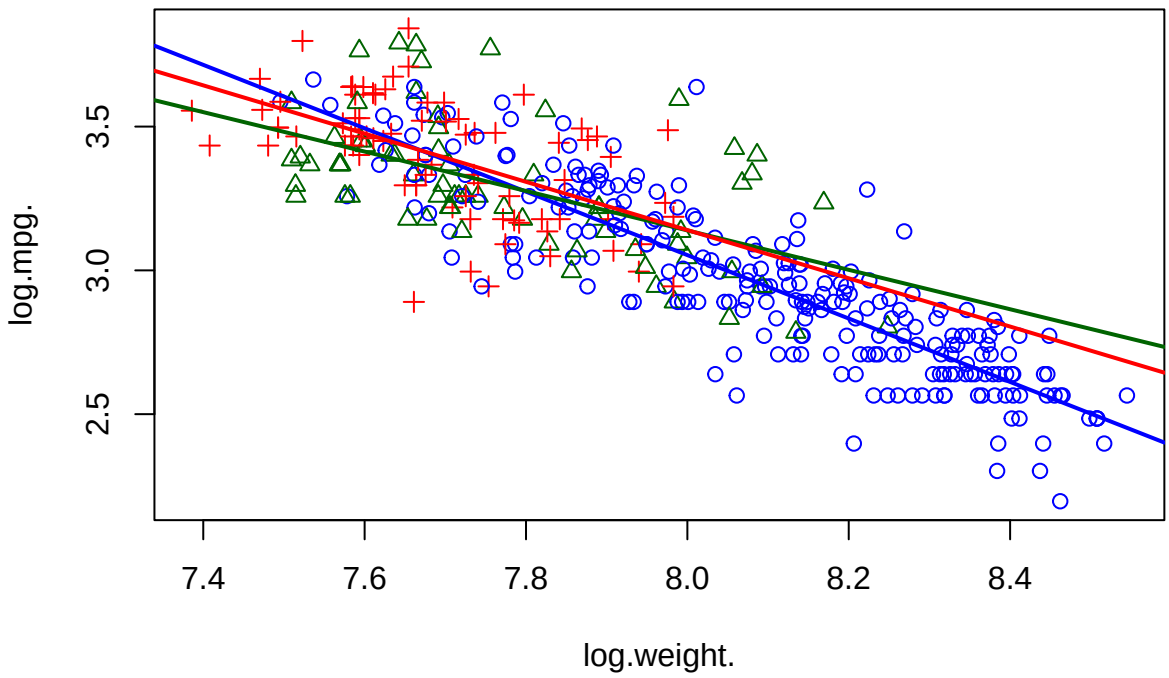
(a) Let's add three separate regression lines on the scatterplot, one for each of the origins: Here's one for the US to get you started:

```
origin_colors = c("blue", "darkgreen", "red")
with(cars_log, plot(log.weight., log.mpg., pch=origin, col=origin_colors[origin]))

cars_us <- subset(cars_log, origin==1)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[1], lwd=2)

cars_us <- subset(cars_log, origin==2)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[2], lwd=2)

cars_us <- subset(cars_log, origin==3)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[3], lwd=2)
```



(b) [not graded] Do cars from different origins appear to have different weight vs. mpg relationships?

ans: No.