Problem Set 4, Fall 2021

Ben Karabinus

```
# Load required packages
library(ggpubr)

## Loading required package: ggplot2

library(leaps)
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.
3.1 ──

## ✓ tibble  3.1.4     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   2.0.1     ✓ forcats 0.5.1
## ✓ purrr   0.3.4

## ── Conflicts ────────────────────────────────────────── tidyverse_conflict
s() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(stringr)
```

CONTEXT: Factorial experiment with doughnuts

Donna is the owner of a boutique doughnut shop. Because many of her customers are conscious of their fat intake but want the flavor of fried doughnuts, she decided to develop a doughnut recipe that minimizes the amount of fat that the doughnuts absorb from the fat in which the doughnuts are fried.

She conducted a factorial experiment that had a similar procedures as Lowe (1935). Like Lowe, she used four types of fats (fat_type). She also used three types of flour (flour_type): all-purpose flour, whole wheat flour, and gluten-free flour. For each combination of fat type and flour type, she cooked six identical batches of doughnuts. Each batch contained 24 doughnuts, and the total fat (in grams) absorbed by the doughnuts in each batch was recorded (sim_tot_fat).

Question 1 - Nested model testing (15 points)

As previously noted, ANOVA is a special case of regression, so anything that can be done in the ANOVA framework can be done in the regression framework. However, this property often isn't obvious when comparing the output of equivalently-specified analyses. For example, the output of the two-way ANOVA with an interaction displays clearly labeled tests of two main effects and one test of the interaction, but the output of the equivalent regression model displays estimates of numerous coefficients that have interpretations different than those used in the ANOVA framework.

In this question, you will use nested model testing to conduct the equivalent tests of main effects and interactions using the regression framework.

Before you start, read in the data and do your data processing.

```
doughnuts.factorial <- read.csv("doughnutsfactorial.csv", header=TRUE, sep=",
") # Loads the CSV file into memory. You may need to adapt this line to work
on your computer
```

As in the previous problem set, please create two new variables in the doughnuts.factorial data set. The first new variable will be called fat_type_factor and will contain the same values as in the fat_type variable but will have a variable type of factor. The second new variable will be called flour_type_factor and will contain the same values as in the flour_type variable but will also have a variable type of factor.

```
# Transform fat_type and flour_type
doughnuts.factorial$fat_type_factor <- as.factor(doughnuts.factorial$fat_type
)

doughnuts.factorial$flour_type_factor <- as.factor(doughnuts.factorial$flour_
type)
```

Check your work by running the following code chunk. Be sure that fat_type_factor and flour_type_factor are factor-type variables before you complete the rest of the problem set.

```
# check the structure of the data set
str(doughnuts.factorial)

## 'data.frame':    72 obs. of  5 variables:
##  $ fat_type         : chr  "Canola" "Canola" "Canola" "Canola" ...
##  $ flour_type       : chr  "ap" "ap" "ap" "ap" ...
##  $ sim_tot_fat      : int  78 71 80 88 62 72 78 75 89 74 ...
##  $ fat_type_factor  : Factor w/ 4 levels "Canola","Peanut",..: 1 1 1 1 1 1
3 3 3 3 ...
##  $ flour_type_factor: Factor w/ 3 levels "ap","gf","ww": 1 1 1 1 1 1 1 1 1
1 ...
```

Question 1, Part 1 - Nested model testing of main effects (5 points)

For this part, you will start by fitting three regression models: an intercept-only model, a fat type-only model, and a flour-type only model. For all models, use sim_tot_fat as the outcome.

Fit the intercept-only model here:

```
# create the null (intercept only) model
model.null <- lm(sim_tot_fat ~ 1, data=doughnuts.factorial)
# print model summary
summary(model.null)

##
## Call:
## lm(formula = sim_tot_fat ~ 1, data = doughnuts.factorial)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.653  -9.903   0.847   9.597  27.347
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   68.653      1.701   40.35   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.44 on 71 degrees of freedom
```

Fit the fat type-only model here:

```
# create the fat_type model
model.fatType <- lm(sim_tot_fat ~ fat_type_factor, data=doughnuts.factorial)
# print the model summary
summary(model.fatType)

##
## Call:
## lm(formula = sim_tot_fat ~ fat_type_factor, data = doughnuts.factorial)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.944  -4.736  -0.167   5.514  21.056
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 66.944      2.529  26.467  < 2e-16 ***
## fat_type_factorPeanut        8.722      3.577   2.438 0.017372 *
## fat_type_factorShortening   11.722      3.577   3.277 0.001654 **
## fat_type_factorSunflower   -13.611      3.577  -3.805 0.000306 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.73 on 68 degrees of freedom
## Multiple R-squared:  0.4708, Adjusted R-squared:  0.4475
## F-statistic: 20.17 on 3 and 68 DF,  p-value: 1.856e-09
```

Fit the flour type-only model here:

```
# create the model for flour_type
model.flourType <- lm(sim_tot_fat ~ flour_type_factor, data=doughnuts.factori
al)
# print model
summary(model.flourType)
```

```
##
## Call:
## lm(formula = sim_tot_fat ~ flour_type_factor, data = doughnuts.factorial)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.083 -11.260   0.563  11.385  29.208
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           74.083      2.880  25.723   <2e-16 ***
## flour_type_factorgf   -8.292      4.073  -2.036   0.0456 *
## flour_type_factorww   -8.000      4.073  -1.964   0.0535 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.11 on 69 degrees of freedom
## Multiple R-squared:  0.07181,    Adjusted R-squared:  0.04491
## F-statistic: 2.669 on 2 and 69 DF,  p-value: 0.07646
```

Now, conduct two nested model tests to conduct the ANOVA-equivalent tests of main effect.

Compare the intercept-only model to the fat type-only model:

```
# nested model test, compare fat_type model to null model
anova(model.null, model.fatType)
```

```
## Analysis of Variance Table
##
## Model 1: sim_tot_fat ~ 1
## Model 2: sim_tot_fat ~ fat_type_factor
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1     71 14798.3
## 2     68  7830.9  3    6967.4 20.167 1.856e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Compare the intercept-only model to the flour type-only model:

```
# compare the flour_type model to the the null model
anova(model.null, model.flourType)

## Analysis of Variance Table
##
## Model 1: sim_tot_fat ~ 1
## Model 2: sim_tot_fat ~ flour_type_factor
##   Res.Df   RSS Df Sum of Sq      F  Pr(>F)
## 1     71 14798
## 2     69 13736  2    1062.7 2.6692 0.07646 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Question 1, Part 2 - Nested model testing of interaction (5 points)

Unlike the previous part, the "base" model for this comparison is not an intercept-only model. Rather, the base model is a model where the interaction is omitted. In the regression framework, this means that the correct reduced model for this ANOVA-equivalent test is a model that includes fat type and flour type, but no interaction between them.

Fit the reduced model, which will contain just fat type and flour type (no interaction), below

```
# create the main model
model.main <- lm(sim_tot_fat ~ fat_type_factor + flour_type_factor,
                 data=doughnuts.factorial)
# print model summary
summary(model.main)

##
## Call:
## lm(formula = sim_tot_fat ~ fat_type_factor + flour_type_factor,
##     data = doughnuts.factorial)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.375  -6.097  -0.229   6.083  23.917
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 72.375      2.923  24.758  < 2e-16 ***
## fat_type_factorPeanut        8.722      3.376   2.584 0.011988 *
## fat_type_factorShortening   11.722      3.376   3.473 0.000914 ***
## fat_type_factorSunflower   -13.611      3.376  -4.032 0.000146 ***
## flour_type_factorgf         -8.292      2.923  -2.836 0.006053 **
## flour_type_factorww         -8.000      2.923  -2.737 0.007967 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.13 on 66 degrees of freedom
## Multiple R-squared:  0.5426, Adjusted R-squared:  0.508
## F-statistic: 15.66 on 5 and 66 DF,  p-value: 3.844e-10
```

Fit the full model, which will contain fat type, flour type, and their interaction

```
# create the full model with interaction
model.interaction <- lm(sim_tot_fat ~ fat_type_factor*flour_type_factor,
                        data=doughnuts.factorial)
# pritn model summary
summary(model.interaction)
```

```
##
## Call:
## lm(formula = sim_tot_fat ~ fat_type_factor * flour_type_factor,
##     data = doughnuts.factorial)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.333  -5.958  -0.250   6.667  21.667
##
## Coefficients:
##                                               Estimate Std. Error t value
## (Intercept)                                     75.167      4.197  17.910
## fat_type_factorPeanut                            3.667      5.935   0.618
## fat_type_factorShortening                        7.167      5.935   1.207
## fat_type_factorSunflower                       -15.167      5.935  -2.555
## flour_type_factorgf                             -8.833      5.935  -1.488
## flour_type_factorww                            -15.833      5.935  -2.668
## fat_type_factorPeanut:flour_type_factorgf        2.333      8.394   0.278
## fat_type_factorShortening:flour_type_factorgf    3.667      8.394   0.437
## fat_type_factorSunflower:flour_type_factorgf    -3.833      8.394  -0.457
## fat_type_factorPeanut:flour_type_factorww       12.833      8.394   1.529
## fat_type_factorShortening:flour_type_factorww   10.000      8.394   1.191
## fat_type_factorSunflower:flour_type_factorww     8.500      8.394   1.013
##                                               Pr(>|t|)
## (Intercept)                                    < 2e-16 ***
## fat_type_factorPeanut                          0.53906
## fat_type_factorShortening                      0.23199
## fat_type_factorSunflower                       0.01316 *
## flour_type_factorgf                            0.14191
## flour_type_factorww                            0.00981 **
## fat_type_factorPeanut:flour_type_factorgf      0.78198
## fat_type_factorShortening:flour_type_factorgf  0.66380
## fat_type_factorSunflower:flour_type_factorgf   0.64954
## fat_type_factorPeanut:flour_type_factorww      0.13154
## fat_type_factorShortening:flour_type_factorww  0.23820
## fat_type_factorSunflower:flour_type_factorww   0.31529
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.28 on 60 degrees of freedom
## Multiple R-squared:  0.5715, Adjusted R-squared:  0.493
## F-statistic: 7.275 on 11 and 60 DF,  p-value: 1.026e-07
```

Now, conduct one nested model test to conduct the ANOVA-equivalent test of the interaction effect.

```
# nested model test for interaction effect
anova(model.main, model.interaction)

## Analysis of Variance Table
##
## Model 1: sim_tot_fat ~ fat_type_factor + flour_type_factor
## Model 2: sim_tot_fat ~ fat_type_factor * flour_type_factor
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     66 6768.3
## 2     60 6340.8  6    427.42 0.6741  0.671
```

Question 1, Part 3 - Interpreting your results (5 points)

You will answer three questions comparing the results of your nested regression model tests and the ANOVA-style tests.

Run the code chunk below to see the results of the one-way ANOVA for fat type you conducted in a previous problem set before answering the question.

```
doughnuts.fat = aov(sim_tot_fat ~ fat_type_factor, data=doughnuts.factorial)

summary(doughnuts.fat)

##                 Df Sum Sq Mean Sq F value   Pr(>F)
## fat_type_factor  3   6967  2322.5   20.17 1.86e-09 ***
## Residuals       68   7831   115.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Look at the results of the nested model test you conducted for *fat type* in *Question 1, Part 1*. Does the F-change test statistic and p-value from that nested model test match the F statistic and p-value (within rounding) of the test in the one-way ANOVA?

Your answer here (yes/no):

Yes

Run the code chunk below to see the results of the one-way ANOVA for flour type you conducted in a previous problem set before answering the question.

```
doughnuts.flour = aov(sim_tot_fat ~ flour_type_factor, data=doughnuts.factori
al)

summary(doughnuts.flour )

##                   Df Sum Sq Mean Sq F value Pr(>F)  
## flour_type_factor  2   1063   531.3   2.669 0.0765 .
## Residuals         69  13736   199.1                 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Look at the results of the nested model test you conducted for *flour type* in *Question 1, Part 1*. Does the F-change test statistic and p-value from that nested model test match the F statistic and p-value (within rounding) of the test in the one-way ANOVA?

Your answer here (yes/no):

Yes

Run the code chunk below to see the results of the two-way ANOVA with an interaction model you conducted in a previous problem set before answering the question.

```
doughnuts.fact.2aov = aov(sim_tot_fat ~ fat_type_factor + flour_type_factor +
fat_type_factor*flour_type_factor, data=doughnuts.factorial)

summary(doughnuts.fact.2aov)

##                                   Df Sum Sq Mean Sq F value   Pr(>F)    
## fat_type_factor                    3   6967  2322.5  21.976 1.01e-09 ***
## flour_type_factor                  2   1063   531.3   5.028  0.00958 ** 
## fat_type_factor:flour_type_factor  6    427    71.2   0.674  0.67095    
## Residuals                         60   6341   105.7                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Look at the results of the nested model test you conducted in *Question 1, Part 2*. Does the F-change test statistic and p-value from the nested model test match the F statistic and p-value (within rounding) of the interaction test in the two-way ANOVA with an interaction?

Your answer here (yes/no):

Yes

CONTEXT - FISHERMAN DATA (many thanks to Dr. Durso for obtaining this data set)

Data Source: N.B. Al-Majed and M.R. Preston (2000). “Factors Influencing the Total Mercury and Methyl Mercury in the Hair of Fishermen in Kuwait,” Environmental Pollution, Vol. 109, pp. 239-250.

http://users.stat.ufl.edu/~winner/datasets.html, downloaded on 4/23/2019

Description: Factors related to mercury levels among fishermen and a control group of non-fishermen.

Variables (names of variables in the data set)

Fisherman indicator (“fisherman”), categorical 0 = No 1 = Yes

Age in years (“age”), continuous

Residence Time in years (“restime”), continuous

Height in cm (“height”), continuous

Weight in kg (“weight”), continuous

Fish meals per week (“fishmlwk”), continuous

Parts of fish consumed (“fishpart”), categorical 0 = none 1 = muscle tissue only 2 = muscle tissue and sometimes whole fish 3 = whole fish

Methyl Mercury in mg/g (“MeHg”), continuous

Total Mercury in mg/g (“TotHg”), continuous

Do this part before starting Questions 2-4!

Before moving on to conducting automated model selection, you'll need to do some data processing. First, set the variables you'll use to the proper data types by completing the lines in the code chunk below. The variables you will include as predictors in your automated model selection are fisherman, age, restime, height, weight, fishmlwk, and fishpart

```
fish <- read.csv("fishermen_mercury.csv", header=TRUE, sep=",") # Loads the C
SV file into memory. You may need to adapt this line to work on your computer

# transform fishpart and fisherman to categorical variables
fish$fisherman_factor <- as.factor(fish$fisherman)

fish$fishpart_factor <- as.factor(fish$fishpart)
```

Check your work by running the following code chunk. Be sure that age, restime, height, weight, and fishmlwk are either integer-type variables or numeric-type variables (R should type these two appropriately automatically) and that fisherman_factor and fishpart_factor are factor-type variables before you complete the rest of the problem set.

```
# check the structure of the data
str(fish)

## 'data.frame':    135 obs. of  11 variables:
##  $ fisherman       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ age             : int  45 38 24 41 43 58 45 46 46 46 ...
##  $ restime         : int  6 13 2 2 11 2 6 0 14 5 ...
##  $ height          : int  175 173 168 183 175 176 184 170 175 175 ...
##  $ weight          : int  70 73 66 80 78 75 85 68 80 75 ...
##  $ fishmlwk        : int  14 7 7 7 21 21 21 7 21 7 ...
##  $ fishpart        : int  2 1 2 1 1 1 1 2 1 1 ...
##  $ MeHg            : num  4.01 4.03 3.58 10.99 10.52 ...
##  $ TotHg           : num  4.48 4.79 3.86 11.44 10.85 ...
##  $ fisherman_factor: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ fishpart_factor : Factor w/ 4 levels "0","1","2","3": 3 2 3 2 2 2 2 3 2
2 ...
```

Next, transform the outcome variable, TotHg, by taking it's log (note: do not conduct a Box Cox transformation; just take the log of the outcome variable). Conducting such a transformation isn't a routine part of automated model selection, but it is an option for improving regression diagnostics and we'll use it for this particular problem.

```
# take natural log of totHg
fish$logTotHg <- log(fish$TotHg)
```

Finally, to make some of the later data manipulation easier, the following code chunk creates a new data set that contains only the transformed outcome and the predictors that will be included in the automated model selection. The select() function as used below requires that you have either the dplyr package or the tidyverse packaged loaded into memory. The first argument identifies the data set (fish) from which variables will be obtained, and the remaining arguments are variables from the fish data set that you want to be copied into the new data set.

```
# create new dataset with for automated model selection
fish.auto <- select(fish, fisherman_factor, fishpart_factor, age, restime, he
ight, weight, fishmlwk, logTotHg)
```

Have one last look at your data structure to check that everything is as expected:

```
# check the data structure
str(fish.auto)

## 'data.frame':    135 obs. of  8 variables:
##  $ fisherman_factor: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ fishpart_factor : Factor w/ 4 levels "0","1","2","3": 3 2 3 2 2 2 2 3 2
2 ...
##  $ age             : int  45 38 24 41 43 58 45 46 46 46 ...
##  $ restime         : int  6 13 2 2 11 2 6 0 14 5 ...
##  $ height          : int  175 173 168 183 175 176 184 170 175 175 ...
##  $ weight          : int  70 73 66 80 78 75 85 68 80 75 ...
##  $ fishmlwk        : int  14 7 7 7 21 21 21 7 21 7 ...
##  $ logTotHg        : num  1.5 1.57 1.35 2.44 2.38 ...
```

Question 2 - Forward selection (10 points)

Use forward selection to find the best set of predictors in the fish.auto data set to predict the log of total mercury (logTotHg). Be sure to include fisherman+_factor, age, restime, height, weight, fishmlwk, and fishpart_factor in your pool of potential predictors. Do not include interaction terms or polynomial terms as part of your pool of potential predictors.

Be sure to include trace=1 in your function.

```
# create formula for forward model selection
fwd_fmla <- as.formula(str_c("logTotHg ~ ",
            str_c(names(fish.auto)[1:(ncol(fish.auto)-1)], collapse = "+")))
# print formula
fwd_fmla

## logTotHg ~ fisherman_factor + fishpart_factor + age + restime +
##     height + weight + fishmlwk

# create the model using automated forward selection
fish.fwd <- step(lm(logTotHg ~1, data=fish.auto), scope = fwd_fmla,
                 direction = "forward", trace = 1)

## Start:  AIC=-35.75
## logTotHg ~ 1
##
```

```
##                      Df Sum of Sq     RSS     AIC
## + weight              1   14.4439  87.622 -54.353
## + fishpart_factor     3   16.2863  85.779 -53.222
## + height              1    2.8525  99.213 -37.580
## + fisherman_factor    1    2.7102  99.356 -37.387
## + fishmlwk            1    2.1889  99.877 -36.680
## <none>                            102.066 -35.754
## + age                 1    0.9863 101.079 -35.065
## + restime             1    0.9818 101.084 -35.059
## 
## Step:  AIC=-54.35
## logTotHg ~ weight
## 
##                      Df Sum of Sq    RSS     AIC
## + fishpart_factor     3   11.0779 76.544 -66.600
## + fisherman_factor    1    3.9920 83.630 -58.648
## + fishmlwk            1    1.7671 85.855 -55.103
## <none>                            87.622 -54.353
## + age                 1    0.6720 86.950 -53.392
## + restime             1    0.3533 87.269 -52.898
## + height              1    0.3217 87.300 -52.849
## 
## Step:  AIC=-66.6
## logTotHg ~ weight + fishpart_factor
## 
##                      Df Sum of Sq    RSS     AIC
## <none>                            76.544 -66.600
## + fisherman_factor    1  0.255106 76.289 -65.051
## + height              1  0.253720 76.290 -65.048
## + age                 1  0.028570 76.515 -64.651
## + fishmlwk            1  0.016532 76.527 -64.629
## + restime             1  0.001631 76.542 -64.603
```

Display the model selected using forward selection by using the summary() function.

```
# print summary  of fish.fwd model
summary(fish.fwd)
```

```
## 
## Call:
## lm(formula = logTotHg ~ weight + fishpart_factor, data = fish.auto)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -4.4462 -0.2406  0.0432  0.4148  1.8856 
## 
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)    
## (Intercept)      -2.94571    0.76915  -3.830 0.000199 ***
## weight            0.04106    0.01037   3.960 0.000123 ***
## fishpart_factor1  1.18211    0.28983   4.079 7.85e-05 ***
## fishpart_factor2  0.99304    0.25679   3.867 0.000173 ***
## fishpart_factor3  1.26124    0.35505   3.552 0.000533 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7673 on 130 degrees of freedom
## Multiple R-squared:  0.2501, Adjusted R-squared:  0.227
## F-statistic: 10.84 on 4 and 130 DF,  p-value: 1.3e-07
```

Question 3 - Backward selection (10 points)

Use backward selection to find the best set of predictors in the fish.auto data set to predict the log of total mercury (logTotHg). Be sure to include fisherman+_factor, age, restime, height, weight, fishmlwk, and fishpart_factor in your pool of potential predictors. Do not include interaction terms or polynomial terms as part of your pool of potential predictors.

Be sure to include trace=1 in your function.

```
# create formula for forward model selection
bwd_fmla <- as.formula(str_c("logTotHg ~ ",
            str_c(names(fish.auto)[1:(ncol(fish.auto)-1)], collapse = "+")))
# create bwd_scope
bwd_scope <- as.formula("logTotHg ~1")
# sanity check
bwd_fmla

## logTotHg ~ fisherman_factor + fishpart_factor + age + restime +
##     height + weight + fishmlwk

bwd_scope

## logTotHg ~ 1

# create the model using automated forward selection
fish.bwd <- step(lm(bwd_fmla, data=fish.auto), scope = bwd_scope,
                 direction = "backward", trace = 1)

## Start:  AIC=-57.62
## logTotHg ~ fisherman_factor + fishpart_factor + age + restime +
##     height + weight + fishmlwk
##
##                    Df Sum of Sq    RSS     AIC
## - restime           1    0.0098 75.981 -59.598
## - age               1    0.0141 75.985 -59.590
## - fishmlwk          1    0.0323 76.003 -59.558
## - height            1    0.2509 76.222 -59.170
## - fisherman_factor  1    0.2666 76.237 -59.142
## <none>                          75.971 -57.615
## - fishpart_factor   3    7.1200 83.091 -51.521
## - weight            1    7.9943 83.965 -46.108
```

```
##
## Step:  AIC=-59.6
## logTotHg ~ fisherman_factor + fishpart_factor + age + height +
##     weight + fishmlwk
##
##                    Df Sum of Sq    RSS     AIC
## - age               1    0.0061 75.987 -61.587
## - fishmlwk          1    0.0299 76.010 -61.545
## - fisherman_factor  1    0.2569 76.237 -61.142
## - height            1    0.2653 76.246 -61.127
## <none>                          75.981 -59.598
## - fishpart_factor   3    7.1148 83.095 -53.514
## - weight            1    8.0450 84.025 -48.011
##
## Step:  AIC=-61.59
## logTotHg ~ fisherman_factor + fishpart_factor + height + weight +
##     fishmlwk
##
##                    Df Sum of Sq    RSS     AIC
## - fishmlwk          1    0.0280 76.015 -63.537
## - height            1    0.2632 76.250 -63.120
## - fisherman_factor  1    0.2767 76.263 -63.096
## <none>                          75.987 -61.587
## - fishpart_factor   3    7.2091 83.196 -55.351
## - weight            1    8.0458 84.032 -50.000
##
## Step:  AIC=-63.54
## logTotHg ~ fisherman_factor + fishpart_factor + height + weight
##
##                    Df Sum of Sq    RSS     AIC
## - height            1    0.2743 76.289 -65.051
## - fisherman_factor  1    0.2756 76.290 -65.048
## <none>                          76.015 -63.537
## - fishpart_factor   3    7.2012 83.216 -57.318
## - weight            1    8.0193 84.034 -51.997
##
## Step:  AIC=-65.05
## logTotHg ~ fisherman_factor + fishpart_factor + weight
##
##                    Df Sum of Sq    RSS     AIC
## - fisherman_factor  1    0.2551 76.544 -66.600
## <none>                          76.289 -65.051
## - fishpart_factor   3    7.3410 83.630 -58.648
## - weight            1    9.4869 85.776 -51.228
##
## Step:  AIC=-66.6
## logTotHg ~ fishpart_factor + weight
##
##                   Df Sum of Sq    RSS     AIC
## <none>                         76.544 -66.600
## - fishpart_factor  3   11.0779 87.622 -54.353
## - weight           1    9.2355 85.779 -53.222
```

Display the model selected using backward selection by using the summary() function.

```
# print model summary
summary(fish.bwd)

## 
## Call:
## lm(formula = logTotHg ~ fishpart_factor + weight, data = fish.auto)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -4.4462 -0.2406  0.0432  0.4148  1.8856 
## 
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)    
## (Intercept)      -2.94571    0.76915  -3.830 0.000199 ***
## fishpart_factor1  1.18211    0.28983   4.079 7.85e-05 ***
## fishpart_factor2  0.99304    0.25679   3.867 0.000173 ***
## fishpart_factor3  1.26124    0.35505   3.552 0.000533 ***
## weight            0.04106    0.01037   3.960 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.7673 on 130 degrees of freedom
## Multiple R-squared:  0.2501, Adjusted R-squared:  0.227 
## F-statistic: 10.84 on 4 and 130 DF,  p-value: 1.3e-07
```

Question 4 - Best subsets selection (10 points)

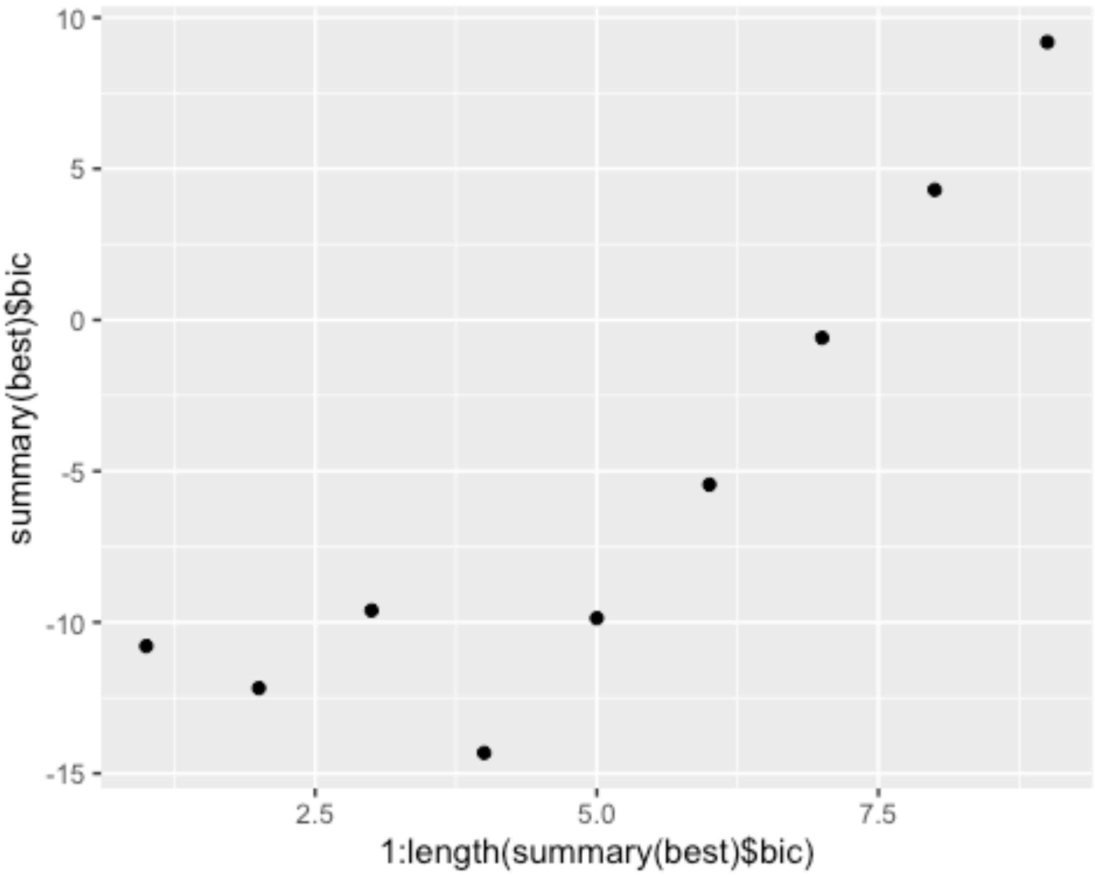
Use best subsets selection to find the best set of predictors in the fish.auto data set to predict the log of total mercury (logTotHg). Be sure to include fisherman_factor, age, restime, height, weight, fishmlwk, and fishpart_factor in your pool of potential predictors. Do not include interaction terms or polynomial terms as part of your pool of potential predictors.

For this problem, choose the best model based on BIC.

```
# create the model matrix "x"
x <- model.matrix(as.formula(str_c("logTotHg ~",
                  str_c(names(fish.auto)[1:ncol(fish.auto)-1],
                        collapse = "+"))),fish.auto)
# create the vector of outcomes
y <- fish.auto$logTotHg

# calculate the best subset of variables - removing (intercept)
best <- regsubsets(x=x[,2:ncol(x)], y=y, method = "exhaustive", nvmax = 9, nb
est=1)
```

Display the model selected using best subsets selection with BIC values.

```
# create the TRUE/FALSE table for the best model
subsetsBoolean <- summary(best)$which
subsetsBoolean

##   (Intercept) fisherman_factor1 fishpart_factor1 fishpart_factor2
## 1        TRUE             FALSE            FALSE            FALSE
## 2        TRUE              TRUE            FALSE            FALSE
## 3        TRUE              TRUE             TRUE            FALSE
## 4        TRUE             FALSE             TRUE             TRUE
## 5        TRUE              TRUE             TRUE             TRUE
## 6        TRUE              TRUE             TRUE             TRUE
## 7        TRUE              TRUE             TRUE             TRUE
## 8        TRUE              TRUE             TRUE             TRUE
## 9        TRUE              TRUE             TRUE             TRUE
##   fishpart_factor3   age restime height weight fishmlwk
## 1            FALSE FALSE   FALSE  FALSE   TRUE    FALSE
## 2            FALSE FALSE   FALSE  FALSE   TRUE    FALSE
## 3            FALSE FALSE   FALSE  FALSE   TRUE    FALSE
## 4             TRUE FALSE   FALSE  FALSE   TRUE    FALSE
## 5             TRUE FALSE   FALSE  FALSE   TRUE    FALSE
## 6             TRUE FALSE   FALSE   TRUE   TRUE    FALSE
## 7             TRUE FALSE   FALSE   TRUE   TRUE     TRUE
## 8             TRUE  TRUE   FALSE   TRUE   TRUE     TRUE
## 9             TRUE  TRUE    TRUE   TRUE   TRUE     TRUE

# display the star table of the best subsets model
subsetsStar <- summary(best)
subsetsStar

## Subset selection object
## 9 Variables  (and intercept)
##                   Forced in Forced out
## fisherman_factor1     FALSE      FALSE
## fishpart_factor1      FALSE      FALSE
## fishpart_factor2      FALSE      FALSE
## fishpart_factor3      FALSE      FALSE
## age                   FALSE      FALSE
## restime               FALSE      FALSE
## height                FALSE      FALSE
## weight                FALSE      FALSE
## fishmlwk              FALSE      FALSE
## 1 subsets of each size up to 9
## Selection Algorithm: exhaustive
##          fisherman_factor1 fishpart_factor1 fishpart_factor2 fishpart_fact
or3
## 1  ( 1 ) " "               " "              " "              " "
## 2  ( 1 ) "*"               " "              " "              " "
## 3  ( 1 ) "*"               "*"              " "              " "
## 4  ( 1 ) " "               "*"              "*"              "*"
## 5  ( 1 ) "*"               "*"              "*"              "*"
## 6  ( 1 ) "*"               "*"              "*"              "*"
## 7  ( 1 ) "*"               "*"              "*"              "*"
## 8  ( 1 ) "*"               "*"              "*"              "*"
## 9  ( 1 ) "*"               "*"              "*"              "*"
##          age restime height weight fishmlwk
## 1  ( 1 ) " " " "     " "    "*"    " "
## 2  ( 1 ) " " " "     " "    "*"    " "
## 3  ( 1 ) " " " "     " "    "*"    " "
## 4  ( 1 ) " " " "     " "    "*"    " "
## 5  ( 1 ) " " " "     " "    "*"    " "
## 6  ( 1 ) " " " "     "*"    "*"    " "
## 7  ( 1 ) " " " "     "*"    "*"    "*"
## 8  ( 1 ) "*" " "     "*"    "*"    "*"
## 9  ( 1 ) "*" "*"     "*"    "*"    "*"

# quick visual of BIC for each subset size
qplot(1:length(summary(best)$bic),summary(best)$bic)
```

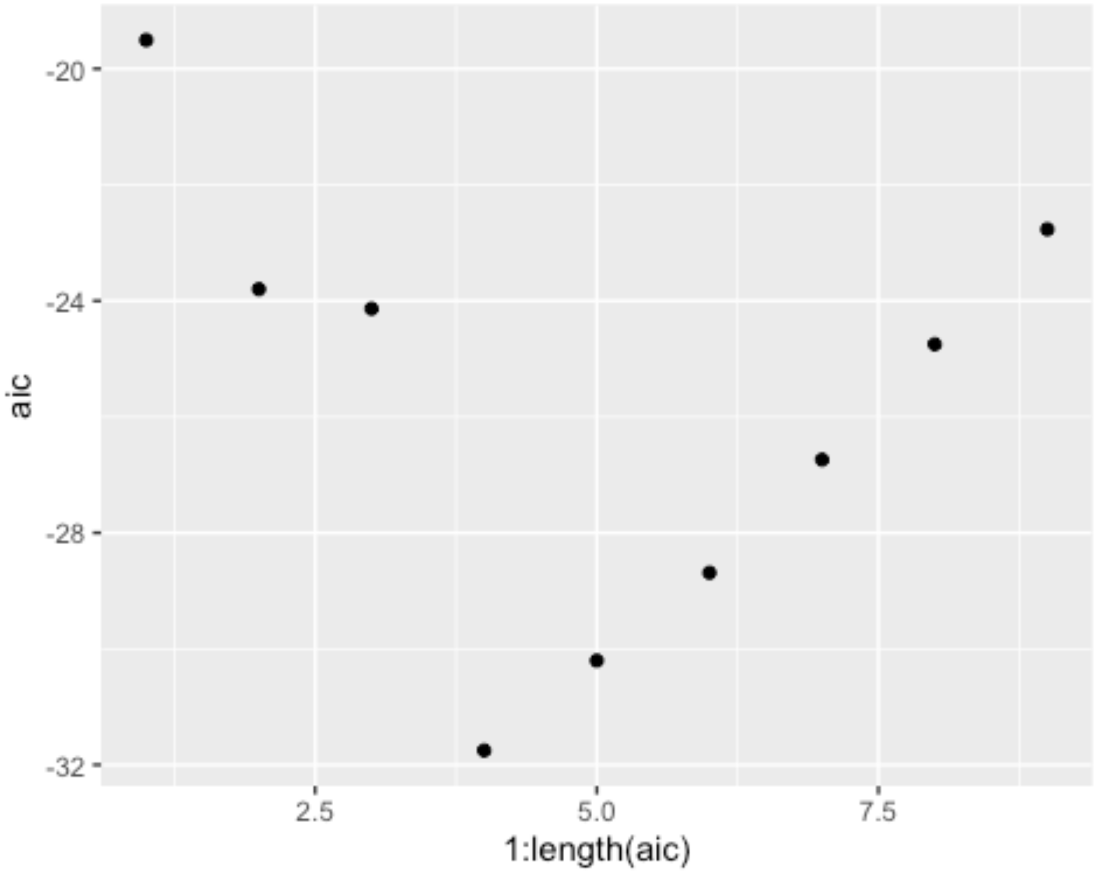


```
# pick the model with the lowest BIC from "best"
# min function searches BIC's and returns the model number model with lowest
BIC
best.subset.bic <- which(summary(best)$bic==min(summary(best)$bic))[1]
```

```
# pull dimension (variable) names from the model based TRUE/FALSE
varnames <- attr(subsetsBoolean, "dimnames")[[2]]

# display the variables in best
best.varnames <- varnames[subsetsBoolean[best.subset.bic,]]

# print variables
best.varnames

## [1] "(Intercept)"      "fishpart_factor1" "fishpart_factor2" "fishpart_fac
tor3"
## [5] "weight"

# create model using variables displayed above
best.lm.bic <- lm(logTotHg ~ fishpart_factor + weight, data=fish.auto)
# print the model summary
summary(best.lm.bic)

##
## Call:
## lm(formula = logTotHg ~ fishpart_factor + weight, data = fish.auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4462 -0.2406  0.0432  0.4148  1.8856
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -2.94571    0.76915  -3.830 0.000199 ***
## fishpart_factor1  1.18211    0.28983   4.079 7.85e-05 ***
## fishpart_factor2  0.99304    0.25679   3.867 0.000173 ***
## fishpart_factor3  1.26124    0.35505   3.552 0.000533 ***
## weight            0.04106    0.01037   3.960 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7673 on 130 degrees of freedom
## Multiple R-squared:  0.2501, Adjusted R-squared:  0.227
## F-statistic: 10.84 on 4 and 130 DF,  p-value: 1.3e-07
```

To compare the results of the best subsets selection with the results of forward and backward selection, you'll need to convert the model BIC values to AIC values. There is an example of how to do this in the async (3.3 Best Subsets Selection).

```
# get bic's
bic <- summary(best)$bic
#  fitted predictors plus intercept and variance
q <- 1:length(bic)+2
# get the number of observations
n <- nrow(fish.auto)
# calculate Akaike information criterion
aic <- bic-log(n)*q+2*q
# quick visual of aic
qplot(1:length(aic),aic)
```



Once this is done, determine the best model using best subsets using the AIC values

```
# extract best subset based on Akaike information criteria
best.subset.aic <- which(aic == min(aic))[1]


# pull model parameters using subsetsBoolean
best.varnames.aic <- varnames[subsetsBoolean[best.subset.aic,]]

# since the same model is produced we can also leverage one of the tables fro
m above
subsetsStar

## Subset selection object
## 9 Variables  (and intercept)
##                   Forced in Forced out
## fisherman_factor1     FALSE      FALSE
## fishpart_factor1      FALSE      FALSE
## fishpart_factor2      FALSE      FALSE
## fishpart_factor3      FALSE      FALSE
## age                   FALSE      FALSE
## restime               FALSE      FALSE
## height                FALSE      FALSE
## weight                FALSE      FALSE
## fishmlwk              FALSE      FALSE
## 1 subsets of each size up to 9
## Selection Algorithm: exhaustive
##          fisherman_factor1 fishpart_factor1 fishpart_factor2 fishpart_fact
or3
## 1  ( 1 ) " "               " "              " "              " "
## 2  ( 1 ) "*"               " "              " "              " "
## 3  ( 1 ) "*"               "*"              " "              " "
## 4  ( 1 ) " "               "*"              "*"              "*"
## 5  ( 1 ) "*"               "*"              "*"              "*"
## 6  ( 1 ) "*"               "*"              "*"              "*"
## 7  ( 1 ) "*"               "*"              "*"              "*"
## 8  ( 1 ) "*"               "*"              "*"              "*"
## 9  ( 1 ) "*"               "*"              "*"              "*"
##          age restime height weight fishmlwk
## 1  ( 1 ) " " " "     " "    "*"    " "
## 2  ( 1 ) " " " "     " "    "*"    " "
## 3  ( 1 ) " " " "     " "    "*"    " "
## 4  ( 1 ) " " " "     " "    "*"    " "
## 5  ( 1 ) " " " "     " "    "*"    " "
## 6  ( 1 ) " " " "     "*"    "*"    " "
## 7  ( 1 ) " " " "     "*"    "*"    "*"
## 8  ( 1 ) "*" " "     "*"    "*"    "*"
## 9  ( 1 ) "*" "*"     "*"    "*"    "*"

# print the variable names
best.varnames.aic
```

```
## [1] "(Intercept)"      "fishpart_factor1" "fishpart_factor2" "fishpart_fac
tor3"
## [5] "weight"
```

Display the model selected using best subsets selection with AIC values.

```
# create the model using the formula for best aic determined above
best.lm.aic <- lm(logTotHg ~ fishpart_factor+weight, data=fish.auto)
summary(best.lm.aic)

##
## Call:
## lm(formula = logTotHg ~ fishpart_factor + weight, data = fish.auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4462 -0.2406  0.0432  0.4148  1.8856
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -2.94571    0.76915  -3.830 0.000199 ***
## fishpart_factor1  1.18211    0.28983   4.079 7.85e-05 ***
## fishpart_factor2  0.99304    0.25679   3.867 0.000173 ***
## fishpart_factor3  1.26124    0.35505   3.552 0.000533 ***
## weight            0.04106    0.01037   3.960 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7673 on 130 degrees of freedom
## Multiple R-squared:  0.2501, Adjusted R-squared:  0.227
## F-statistic: 10.84 on 4 and 130 DF,  p-value: 1.3e-07
```

Question 5 - 5 points

Question 1: Which predictors were included in the model you chose using forward selection?

Your answer here:

The predictors “weight” and “fishpart_factor” were chosen by the forward selection model.

Question 2: Which predictors were included in the model you chose using backward selection?

Your answer here:

The predictors “weight” and “fishpart_factor” were chosen by the model using backward selection.

Question 3: Which predictors were included in the model you chose using best subsets selection (AIC)?

Your answer here:

The predictors “weight and”fishpart_factor” were chosen by the best subsets model using AIC as the performance metric.

Question 4: Which predictors were included in the model you chose using best subsets selection (BIC)?

Your answer here:

The predictors “weight and”fishpart_factor” were chosen by the best subsets model using BIC as the performance metric.