Problem Set 4, Winter 2021

Adam Ten Hoeve

```r
# Load any packages, if any, that you use as part of your answers here
# For example:
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggpubr)
library(leaps)
```

CONTEXT - DOUGHNUTS DATA

As a reminder, I decided to conduct a factorial experiment inspired by the experiment conducted by Lowe (1935) to learn more about how much fat doughnuts absorb in different conditions. Like Lowe, I used four types of fats (fat_type). I also used three types of flour (flour_type): all-purpose flour, whole wheat flour, and gluten-free flour. Again like Lowe, I cooked six identical batches of doughnuts in each flour and fat combination. 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 some way in the regression framework. When conducting a two-way factorial ANOVA, you can test for main effects and the interaction between the two variables. When you coded this as a regression in the previous problem set, you ended up with a model with many coefficients associated with the interaction. You can, however, do an ANOVA-style all-at-once test of the interaction using nested model testing.

First, load the data into memory and make the appropriate changes to the variables.

```r
# Code for loading and setting up your data appropriately.
doughnuts = read.csv("doughnutsfactorial.csv")
doughnuts$fat_type = as.factor(doughnuts$fat_type)
```

```
doughnuts$flour_type = as.factor(doughnuts$flour_type)

# Don’t forget to display using the str() function!
str(doughnuts)
```

```
## ’data.frame’:    72 obs. of  3 variables:
##  $ fat_type   : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 2 2 2 2 ...
##  $ flour_type : Factor w/ 3 levels "ap","gf","ww": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sim_tot_fat: int  78 71 80 88 62 72 78 75 89 74 ...
```

Next, specify your two regression models. The first model will have just the vectors associated with main effects, and the second model will have both the main effects and interaction vectors. Please display the results of both using the summary() function.

```
# Code for your regression models
doughnuts.lmod.1 = lm(sim_tot_fat~fat_type+flour_type, data=doughnuts)
summary(doughnuts.lmod.1)
```

```
##
## Call:
## lm(formula = sim_tot_fat ~ fat_type + flour_type, data = doughnuts)
##
## 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_type2      11.722      3.376   3.473 0.000914 ***
## fat_type3       8.722      3.376   2.584 0.011988 *
## fat_type4     -13.611      3.376  -4.032 0.000146 ***
## flour_typegf   -8.292      2.923  -2.836 0.006053 **
## flour_typeww   -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
```

```
# Use the summary() function to display your results!
doughnuts.lmod.2 = lm(sim_tot_fat ~ fat_type*flour_type, data=doughnuts)
summary(doughnuts.lmod.2)
```

```
##
## Call:
## lm(formula = sim_tot_fat ~ fat_type * flour_type, data = doughnuts)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.333  -5.958  -0.250   6.667  21.667 
## 
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)    
## (Intercept)               75.167      4.197  17.910  < 2e-16 ***
## fat_type2                  7.167      5.935   1.207  0.23199    
## fat_type3                  3.667      5.935   0.618  0.53906    
## fat_type4                -15.167      5.935  -2.555  0.01316 *  
## flour_typegf              -8.833      5.935  -1.488  0.14191    
## flour_typeww             -15.833      5.935  -2.668  0.00981 ** 
## fat_type2:flour_typegf     3.667      8.394   0.437  0.66380    
## fat_type3:flour_typegf     2.333      8.394   0.278  0.78198    
## fat_type4:flour_typegf    -3.833      8.394  -0.457  0.64954    
## fat_type2:flour_typeww    10.000      8.394   1.191  0.23820    
## fat_type3:flour_typeww    12.833      8.394   1.529  0.13154    
## fat_type4:flour_typeww     8.500      8.394   1.013  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
```

Finally, conduct the F-change test to determine if the interaction is significant and state what conclusion you reach (hint: make sure your degrees of freedom are positive):

```r
# Code for F-change test
anova(doughnuts.lmod.1, doughnuts.lmod.2)
```

```
## Analysis of Variance Table
## 
## Model 1: sim_tot_fat ~ fat_type + flour_type
## Model 2: sim_tot_fat ~ fat_type * flour_type
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     66 6768.2                           
## 2     60 6340.8  6    427.42 0.6741  0.671
```

Write your conclusion about the significance of the interaction here:

From the nested-model test, we get a p-value of 0.671. This is a large value, so we fail to reject the null hypothesis of the test. Therefor, there is not a significant difference between the models. This means that the interaction term is not significant because is didn’t significantly improve the fit of the model.

CONTEXT - FISHERMAN DATA (adapted from Cathy Durso’s material)

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)

Age in years (age)

Residence Time in years (restime)

Height in cm (height)

Weight in kg (weight)

Fish meals per week (fishmlwk)

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

Methyl Mercury in mg/g (MeHg)

Total Mercury in mg/g (TotHg)

Question 2 - Forward selection (10 points)

Use forward selection to find the best set of predictors to predict the log of total mercury. Be sure to include fisherman, age, restime, height, weight, fishmlwk, and fishpart in your pool of potential predictors. Note that fishpart and fisherman should be categorical variables. Do not include MeHg in your set of predictors.

First, load the data into memory and change the variable types as appropriate. Please show the structure of your data in your knitted document by using the str() function.

```
# Code for loading and setting up your data appropriately. These changes will apply to the next two que
fishermen = read.csv("fishermen_mercury.csv")
fishermen$fisherman = as.factor(fishermen$fisherman)
fishermen$fishpart = as.factor(fishermen$fishpart)

# Don't forget to display using the str() function!
str(fishermen)
```

```
## 'data.frame':    135 obs. of  9 variables:
##  $ fisherman: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 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 ...
##  $ fishpart : Factor w/ 4 levels "0","1","2","3": 3 2 3 2 2 2 2 3 2 2 ...
##  $ MeHg     : num  4.01 4.03 3.58 10.99 10.52 ...
##  $ TotHg    : num  4.48 4.79 3.86 11.44 10.85 ...
```

Next, conduct your forward selection. Be sure to include trace=ON in your function.

```
# Code for conduting a forward selection
# Specify the maximum scope model
full.model.formula = as.formula("log(TotHg)~fisherman+age+restime+height+weight+fishmlwk+fishpart")

# Perform formward selection with the step() function
fisherman.lmod.forward = step(lm(log(TotHg)~1, data=fishermen),
                              scope=full.model.formula, direction="forward", trace=1)
```

```
## Start:  AIC=-35.75
## log(TotHg) ~ 1
##
##              Df Sum of Sq     RSS     AIC
## + weight     1   14.4439  87.622 -54.353
## + fishpart   3   16.2863  85.779 -53.222
## + height     1    2.8525  99.213 -37.580
## + fisherman  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
## log(TotHg) ~ weight
##
##              Df Sum of Sq    RSS     AIC
## + fishpart   3   11.0779 76.544 -66.600
## + fisherman  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
## log(TotHg) ~ weight + fishpart
##
##              Df Sum of Sq    RSS     AIC
## <none>                   76.544 -66.600
## + fisherman  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
```

Finally, display the final model using the summary() function.

```
# Display the model selected by forward selection using the summary() function!
summary(fisherman.lmod.forward)
```

```
##
## Call:
## lm(formula = log(TotHg) ~ weight + fishpart, data = fishermen)
##
## 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 ***
## fishpart1    1.18211    0.28983   4.079 7.85e-05 ***
## fishpart2    0.99304    0.25679   3.867 0.000173 ***
## fishpart3    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 to predict the log of total mercury. Be sure to include fisherman, age, restime, height, weight, fishmlwk, and fishpart in your pool of potential predictors. Note that fishpart and fisherman should be categorical variables. Do not include MeHg in your set of predictors.

First, conduct your backward selection. Be sure to include trace=ON in your function.

```
# Code for conduting a backward selection
# Specify the minimum model formula
model.min.formula = as.formula("log(TotHg)~1")
# Perform backward selection using step()
fishermen.lmod.backward = step(lm(full.model.formula, data=fishermen),
                               scope=model.min.formula, direction="backward", trace=1)
```

```
## Start:  AIC=-57.62
## log(TotHg) ~ fisherman + age + restime + height + weight + fishmlwk +
##     fishpart
##
##             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  1    0.2666 76.237 -59.142
## <none>                   75.971 -57.615
## - fishpart   3    7.1200 83.091 -51.521
## - weight     1    7.9943 83.965 -46.108
##
## Step:  AIC=-59.6
## log(TotHg) ~ fisherman + age + height + weight + fishmlwk + fishpart
##
##             Df Sum of Sq    RSS     AIC
## - age        1    0.0061 75.987 -61.587
## - fishmlwk   1    0.0299 76.010 -61.545
## - fisherman  1    0.2569 76.237 -61.142
## - height     1    0.2653 76.246 -61.127
## <none>                   75.981 -59.598
## - fishpart   3    7.1148 83.095 -53.514
## - weight     1    8.0450 84.025 -48.011
##
## Step:  AIC=-61.59
## log(TotHg) ~ fisherman + height + weight + fishmlwk + fishpart
```

```
##
##             Df Sum of Sq    RSS     AIC
## - fishmlwk   1    0.0280 76.015 -63.537
## - height     1    0.2632 76.250 -63.120
## - fisherman  1    0.2767 76.263 -63.096
## <none>                   75.987 -61.587
## - fishpart   3    7.2091 83.196 -55.351
## - weight     1    8.0458 84.032 -50.000
##
## Step:  AIC=-63.54
## log(TotHg) ~ fisherman + height + weight + fishpart
##
##             Df Sum of Sq    RSS     AIC
## - height     1    0.2743 76.289 -65.051
## - fisherman  1    0.2756 76.290 -65.048
## <none>                   76.015 -63.537
## - fishpart   3    7.2012 83.216 -57.318
## - weight     1    8.0193 84.034 -51.997
##
## Step:  AIC=-65.05
## log(TotHg) ~ fisherman + weight + fishpart
##
##             Df Sum of Sq    RSS     AIC
## - fisherman  1    0.2551 76.544 -66.600
## <none>                   76.289 -65.051
## - fishpart   3    7.3410 83.630 -58.648
## - weight     1    9.4869 85.776 -51.228
##
## Step:  AIC=-66.6
## log(TotHg) ~ weight + fishpart
##
##            Df Sum of Sq    RSS     AIC
## <none>                  76.544 -66.600
## - fishpart  3   11.0779 87.622 -54.353
## - weight    1    9.2355 85.779 -53.222
```

Next, display the final model using the summary() function.

```
# Display the model selected by backward selection using the summary() function!
summary(fishermen.lmod.backward)
```

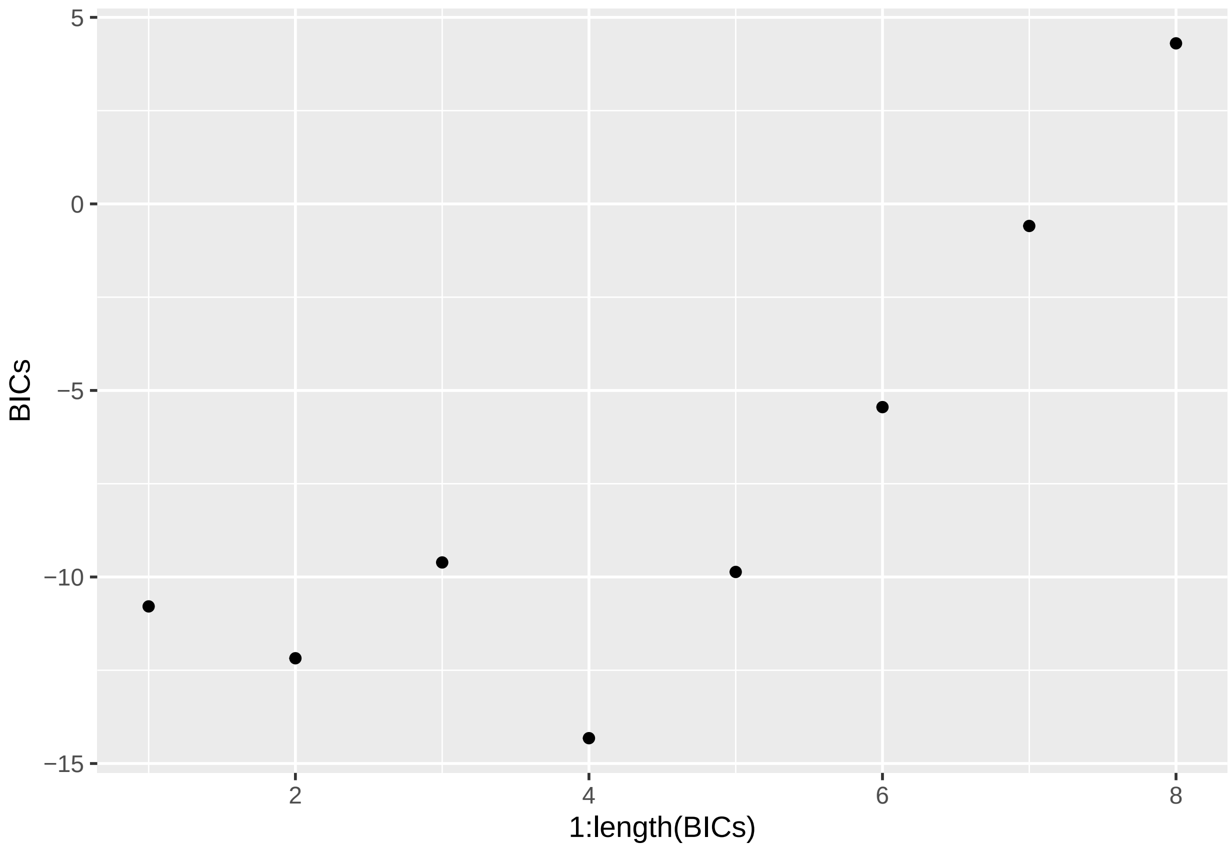
```
##
## Call:
## lm(formula = log(TotHg) ~ weight + fishpart, data = fishermen)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4462 -0.2406  0.0432  0.4148  1.8856
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.94571    0.76915  -3.830 0.000199 ***
## weight       0.04106    0.01037   3.960 0.000123 ***
## fishpart1    1.18211    0.28983   4.079 7.85e-05 ***
## fishpart2    0.99304    0.25679   3.867 0.000173 ***
## fishpart3    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 4 - Best subsets selection (10 points)

Use best subsets selection to find the best set of predictors to predict the log of total mercury. Be sure to include fisherman, age, restime, height, weight, fishmlwk, and fishpart in your pool of potential predictors. Note that fishpart and fisherman should be categorical variables. Do not include MeHg in your set of predictors.

First, conduct your best subsets selection.

```r
# Code for conduting a best subsets selection
# Find the best subsets of for each number of predictors
x = model.matrix(full.model.formula, fishermen)
best.subsets = regsubsets(x=x[, 2:ncol(x)], y=log(fishermen$TotHg),
                          method="exhaustive", nvmax=8, nbest=1)
subsets = summary(best.subsets)$which
# Calculate the BIC value for each of those subsets. Plot the results.
BICs = summary(best.subsets)$bic
qplot(1:length(BICs), BICs)
```



```
# From the graph, we see that the lowest BIC occurs when there are 4 predictors
subsets
```

```
##   (Intercept) fisherman1   age restime height weight fishmlwk fishpart1
## 1        TRUE      FALSE FALSE   FALSE  FALSE   TRUE    FALSE     FALSE
## 2        TRUE       TRUE FALSE   FALSE  FALSE   TRUE    FALSE     FALSE
## 3        TRUE       TRUE FALSE   FALSE  FALSE   TRUE    FALSE      TRUE
## 4        TRUE      FALSE FALSE   FALSE  FALSE   TRUE    FALSE      TRUE
## 5        TRUE       TRUE FALSE   FALSE  FALSE   TRUE    FALSE      TRUE
## 6        TRUE       TRUE FALSE   FALSE   TRUE   TRUE    FALSE      TRUE
## 7        TRUE       TRUE FALSE   FALSE   TRUE   TRUE     TRUE      TRUE
## 8        TRUE       TRUE  TRUE   FALSE   TRUE   TRUE     TRUE      TRUE
##   fishpart2 fishpart3
## 1     FALSE     FALSE
## 2     FALSE     FALSE
## 3     FALSE     FALSE
## 4      TRUE      TRUE
## 5      TRUE      TRUE
## 6      TRUE      TRUE
## 7      TRUE      TRUE
## 8      TRUE      TRUE
```

```
# We see that the predictors for the best subsets model are weight and fishpart.
fishermen.best.subset = lm(log(TotHg)~weight+fishpart, data=fishermen)
```

Next, display the final model using the summary() function.

```r
# use the summary() function!
summary(fishermen.best.subset)
```

```
##
## Call:
## lm(formula = log(TotHg) ~ weight + fishpart, data = fishermen)
##
## 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 ***
## fishpart1    1.18211    0.28983   4.079 7.85e-05 ***
## fishpart2    0.99304    0.25679   3.867 0.000173 ***
## fishpart3    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 5 - 5 points

Were there any differences between the models chosen by the three different automated model selection methods? If so, how did they differ?

Your answer here: All three of the model selection techniques resulted in the same selection of predictors, being only `weight` and `fishpart`.