C1M3_autograded

December 23, 2021

1 Module 3 - Autograded Assignment

1.0.1 Outline:

Here are the objectives of this assignment:

1. Utilize F-tests to distinguish between statistically different models.
2. Calculate Confidence Intervals for feature parameters to understand their variability.
3. Reinforce an understanding of Confidence Intervals by comparing many different CIs from the same underlying population.
4. Improve general familiarity with R, including utilizing data frames and ggplot.

Here are some general tips:

1. Read the questions carefully to understand what is being asked.
2. When you feel that your work is completed, feel free to hit the `Validate` button to see your results on the *visible* unit tests. If you have questions about unit testing, please refer to the "Module 0: Introduction" notebook provided as an optional resource for this course. In this assignment, there are hidden unit tests that check your code. You will not recieve any feedback for failed hidden unit tests until the assignment is submitted. **Do not misinterpret the feedback from visible unit tests as all possible tests for a given question–write your code carefully!**
3. Before submitting, we recommend restarting the kernel and running all the cells in order that they appear to make sure that there are no additional bugs in your code.
4. There are 50 points total in this assignment.

[1]:
```
# This cell loads the necesary libraries for this assignment
library(testthat)
library(tidyverse)
library(RCurl)  # a package that includes the function getURL(), which allows␣
 ↪for reading data from github.
library(ggplot2)
```

```
Error in get(genname, envir = envir) : object 'testthat_print' not found

  Attaching packages                                  tidyverse
1.3.0

 ggplot2 3.3.0       purrr    0.3.4
 tibble  3.0.1       dplyr    0.8.5
```

```
 tidyr   1.0.2      stringr 1.4.0
 readr   1.3.1      forcats 0.5.0

  Conflicts
tidyverse_conflicts()
 dplyr::filter()  masks stats::filter()
 purrr::is_null() masks
testthat::is_null()
 dplyr::lag()     masks stats::lag()
 dplyr::matches() masks
tidyr::matches(), testthat::matches()


Attaching package: 'RCurl'


The following object is masked from 'package:tidyr':

    complete
```

2 Problem 1: Comparing Models

In this exercise, we will fit multiple different models to the same data and determine which of those models we should ultimately use.

The data we will be using is the Auto MPG Data Set from the UCI Machine Learning Repository. It contains technical specifications and performance ratings of many different cars. We will focus on the features that impact the overall `mpg` of each car.

In the cell below, code is provided for you to load in the data and rename the columns to be more specific.

[2]:
```r
mpg.data = read_table("auto-mpg.data")
names(mpg.data) = c("mpg", "cylinders", "displacement", "horsepower", "weight",
                    "accel", "model_year", "origin", "car_name")
mpg.data$horsepower = as.numeric(mpg.data$horsepower)
mpg.data = na.omit(mpg.data)

summary(mpg.data)
str(mpg.data)
head(mpg.data)
```

```
Parsed with column specification:
cols(
  `18.0` = col_double(),
  `8` = col_double(),
```

```
  `307.0` = col_double(),
  `130.0` = col_character(),
  `3504.` = col_double(),
  `12.0` = col_double(),
  `70` = col_double(),
  `1` = col_double(),
  `"chevrolet chevelle malibu"` = col_character()
)
```

```
Warning message in eval(expr, envir, enclos):
"NAs introduced by coercion"
```

```
      mpg          cylinders      displacement     horsepower        weight
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2224
 Median :23.00   Median :4.000   Median :151.0   Median : 93.0   Median :2800
 Mean   :23.46   Mean   :5.465   Mean   :194.1   Mean   :104.4   Mean   :2976
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:264.5   3rd Qu.:125.0   3rd Qu.:3616
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
     accel         model_year        origin        car_name
 Min.   : 8.00   Min.   :70.00   Min.   :1.000   Length:391
 1st Qu.:13.80   1st Qu.:73.00   1st Qu.:1.000   Class :character
 Median :15.50   Median :76.00   Median :1.000   Mode  :character
 Mean   :15.55   Mean   :75.99   Mean   :1.578
 3rd Qu.:17.05   3rd Qu.:79.00   3rd Qu.:2.000
 Max.   :24.80   Max.   :82.00   Max.   :3.000
```

```
tibble [391 × 9] (S3: tbl_df/tbl/data.frame)
 $ mpg         : num [1:391] 15 18 16 17 15 14 14 14 15 15 …
 $ cylinders   : num [1:391] 8 8 8 8 8 8 8 8 8 8 …
 $ displacement: num [1:391] 350 318 304 302 429 454 440 455 390 383 …
 $ horsepower  : num [1:391] 165 150 150 140 198 220 215 225 190 170 …
 $ weight      : num [1:391] 3693 3436 3433 3449 4341 …
 $ accel       : num [1:391] 11.5 11 12 10.5 10 9 8.5 10 8.5 10 …
 $ model_year  : num [1:391] 70 70 70 70 70 70 70 70 70 70 …
 $ origin      : num [1:391] 1 1 1 1 1 1 1 1 1 1 …
 $ car_name    : chr [1:391] "\"buick skylark 320\"" "\"plymouth satellite\""
"\"amc rebel sst\"" "\"ford torino\"" …
 - attr(*, "na.action")= 'omit' Named int [1:6] 32 126 330 336 354 374
  ..- attr(*, "names")= chr [1:6] "32" "126" "330" "336" …
```

A tibble: 6 × 9

mpg <dbl>	cylinders <dbl>	displacement <dbl>	horsepower <dbl>	weight <dbl>	accel <dbl>	model_year <dbl>	origin <dbl>	car_r <chr
15	8	350	165	3693	11.5	70	1	"buic
18	8	318	150	3436	11.0	70	1	"plyr
16	8	304	150	3433	12.0	70	1	"amc
17	8	302	140	3449	10.5	70	1	"ford
15	8	429	198	4341	10.0	70	1	"ford
14	8	454	220	4354	9.0	70	1	"chev

1. (a) Three Different Models (5 points) We will fit three different models to this data:

1. `mod.1`: Fits `mpg` as the response with `weight` as the predictor.
2. `mod.2`: Fits `mpg` as the response with `weight` and `accel` as predictors.
3. `mod.3`: Fits `mpg` as the response with `weight`, `accel` and `horsepower` as predictors.

Fit these models in the cell below.

[3]:
```
mod.1 = NA
mod.2 = NA
mod.3 = NA
# your code here
mod.1 = lm(mpg~weight, data=mpg.data)
mod.2 = lm(mpg~weight + accel, data=mpg.data)
mod.3 = lm(mpg ~ weight + accel + horsepower, data=mpg.data)
```

[4]:
```
# Test Cell
# Make sure that each model is a linear model
if(test_that("Testing model types",
             {(expect_is(mod.1, "lm"))
              (expect_is(mod.2, "lm"))
              (expect_is(mod.3, "lm"))})){
    print("All models are linear models.")
}else{
    print("At least one of the models isn't a linear model!")
    print("Make sure you're using the lm() function.")
}
# This cell has hidden test cases that will run after submission.
```

```
[1] "All models are linear models."
```

1. (b) Partial F-Tests (10 points) Compare the 3 models using pairwise F-tests to determine which of the three we should use moving forward. It may be helpful to write out the null and alternative hypotheses for these tests.

Copy your selected model into the `final.model` variable.

[5]:
```
#mod.1 model sumarry
summary(mod.1)
```

```
Call:
lm(formula = mpg ~ weight, data = mpg.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-11.9749  -2.7599  -0.3187   2.1423  16.5180 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 46.2122411  0.7996945   57.79   <2e-16 ***
weight      -0.0076447  0.0002584  -29.59   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.338 on 389 degrees of freedom
Multiple R-squared:  0.6923,Adjusted R-squared:  0.6915 
F-statistic: 875.3 on 1 and 389 DF,  p-value: < 2.2e-16
```

[6]:
```
#mod.2 model sumarry
summary(mod.2)
```

```
Call:
lm(formula = mpg ~ weight + accel, data = mpg.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-11.1402  -2.7879  -0.3357   2.4250  16.2099 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 41.1086887  1.8722680  21.957   <2e-16 ***
weight      -0.0072929  0.0002812 -25.932   <2e-16 ***
accel        0.2608627  0.0867257   3.008   0.0028 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.293 on 388 degrees of freedom
Multiple R-squared:  0.6993,Adjusted R-squared:  0.6978 
F-statistic: 451.3 on 2 and 388 DF,  p-value: < 2.2e-16
```

[7]:
```
#mod.3 model sumarry
summary(mod.3)
```

```
Call:
```

```
lm(formula = mpg ~ weight + accel + horsepower, data = mpg.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-11.0845  -2.7434  -0.3301   2.1861  16.2623 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 45.7195472  2.4158434  18.925   <2e-16 ***
weight      -0.0057832  0.0005787  -9.994   <2e-16 ***
accel       -0.0044729  0.1237793  -0.036   0.9712    
horsepower  -0.0476799  0.0160215  -2.976   0.0031 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.251 on 387 degrees of freedom
Multiple R-squared:  0.7061,Adjusted R-squared:  0.7038 
F-statistic: 309.9 on 3 and 387 DF,  p-value: < 2.2e-16
```

Model 3 is our full model. Doing the full model test, we see there needs to be at least one predictor in the model.

For the following below we have:
H_0: the reduced model of only `weight` should be included.
H_1: More than `weight` should be included as variables should be included in the model.

[8]:
```
#mod1 vs mod2
anova(mod.1, mod.2)
# need more than just the reduced model (mod.1)
```

A anova: 2 × 6

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	389	7319.209	NA	NA	NA	NA
2	388	7152.427	1	166.7822	9.047486	0.002801995

[9]:
```
#mod1 vs mod3
anova(mod.1, mod.3)
# need more than just the reduced model (mod.1)
```

A anova: 2 × 6

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	389	7319.209	NA	NA	NA	NA
2	387	6992.405	2	326.8045	9.043622	0.0001450187

[10]:
```
#mod2 vs mod3
anova(mod.2, mod.3)
# need more than just the reduced model (mod.2)
```

A anova: 2 × 6

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	388	7152.427	NA	NA	NA	NA
2	387	6992.405	1	160.0223	8.856557	0.003103755

[11]:
```
final.model = NA
# your code here
final.model = mod.3
```

[12]:
```
# Test Cell
if(test_that("Check final.model class", {expect_is(final.model, "lm")})){
    print("You've selected a model! Make sure you're confident in your answer.")
}else{
    print("final.model is not a linear model.")
    print("To copy the selected model use `final.model = mod.#`")
}
# This cell has hidden test cases that will run after submission.
```

```
[1] "You've selected a model! Make sure you're confident in your answer."
```

1. (c) Coefficient Confidence Intervals (10 points) Using your selected best model, calculate a 95% confidence interval for the `weight` parameter. Save the lower and upper values into `weight.CI.lower` and `weight.CI.upper` respectively.

[13]:
```
weight.CI.lower = NA
weight.CI.upper = NA

# your code here
confint(mod.3)
weight.CI.lower = confint(mod.3)[2,1]
weight.CI.upper = confint(mod.3)[2,2]
```

A matrix: 4 × 2 of type dbl

	2.5 %	97.5 %
(Intercept)	40.969726622	50.469367698
weight	-0.006920931	-0.004645464
accel	-0.247836965	0.238891259
horsepower	-0.079179930	-0.016179840

[14]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

1. (d) Model Comparison (5 points) So far, we've used the F-test as a way to choose a "best" model among the three proposed. Now let's compare the models according to their mean squared errors (MSE). Compute the MSE for each of the three models and save their values into their respective `MSE.#` variables.

Which of these models has the best MSE? Do these conclusions agree with the model you selected in part **1.b**? Think about why or why not.

[15]:
```
MSE.1 = NA
MSE.2 = NA
MSE.3 = NA

# your code here
MSE.1 = mean(summary(mod.1)$residuals^2)
MSE.2 = mean(summary(mod.2)$residuals^2)
MSE.3 = mean(summary(mod.3)$residuals^2)
MSE.1
MSE.2
MSE.3
```

18.7192058382486

18.2926529344723

17.8833887378182

[16]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

3 Problem 2: Large Datasets and Significance

For this exercise, we will see if we can create a "good" regression model for a city's temperature using other weather data. The data is from hourly weather records of Szeged, Hungary from 2006-2016. The data was provided by Darksky.net and can be found on Kaggle here. The data has not been modified in any way.

The data is loaded in the cell below.

[17]:
```
# Load in the data
weather.data = read.csv("weatherHistory.csv")
weather.data = na.omit(weather.data)
head(weather.data)
```

A data.frame: 6 × 12

	Formatted.Date <fct>	Summary <fct>	Precip.Type <fct>	Temperature..C. <dbl>	Ap <d
1	2006-04-01 00:00:00.000 +0200	Partly Cloudy	rain	9.472222	7.3
2	2006-04-01 01:00:00.000 +0200	Partly Cloudy	rain	9.355556	7.2
3	2006-04-01 02:00:00.000 +0200	Mostly Cloudy	rain	9.377778	9.3
4	2006-04-01 03:00:00.000 +0200	Partly Cloudy	rain	8.288889	5.9
5	2006-04-01 04:00:00.000 +0200	Mostly Cloudy	rain	8.755556	6.9
6	2006-04-01 05:00:00.000 +0200	Partly Cloudy	rain	9.222222	7.1

2. (a) Talking about the weather. (5 points) Before we jump into modeling, let's think about weather. Is temperature correlated with wind speed, visibility or pressure? Certainly somewhat, but probably not to a great extent. Let's find out exactly (at least for these data).

Determine the correlation between `Temperature..C.` and the three predictors: `Wind.Speed..km.h.`, `Visibility..km.` and `Pressure..millibars..` Store these values in `cor.speed`, `cor.vis` and `cor.pres` respectively.

Also, if our data is hourly records over 10 years, then we're going to have a lot of records. How many rows does our dataset have? Store this value in `data.n`.

[18]:
```
cor.speed = NA
cor.vis = NA
cor.pres = NA
data.n = NA

# your code here
cor.speed = cor(weather.data$Temperature..C., weather.data$Wind.Speed..km.h)
cor.vis = cor(weather.data$Temperature..C., weather.data$Visibility..km.)
cor.pres = cor(weather.data$Temperature..C., weather.data$Pressure..millibars.)
data.n = nrow(weather.data)
cor.speed
cor.vis
cor.pres
data.n
```

0.0214397879642656

0.376669738692084

-0.0387464175102724

54919

[19]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2. (b) Data Size Matters (5 points) Yep, that's a lot of data. But isn't more data better? Well, let's find out. We can create two different models, one with a little data and one with a lot of data, and determine if the one fit to more data is the better model.

Fit two models to the data, with `Temperature..C.` as the response and `Wind.Speed..km.h.`, `Visibility..km.` and `Pressure..millibars.` as predictors. The first model, `weather.lmod.small`, should be fit to the first 30 rows of the data. The second model, `weather.lmod.all`, should be fit to all the data.

Look at the p-values of the model coefficients. What can you infer?

[20]:
```
weather.lmod.small = NA
weather.lmod.all = NA
```

```r
# your code here
weather.lmod.small = lm(Temperature..C. ~ Wind.Speed..km.h. + Visibility..km. +
                        Pressure..millibars., data=weather.data[1:30,])

weather.lmod.all = lm(Temperature..C. ~ Wind.Speed..km.h. + Visibility..km. +
                        Pressure..millibars., data=weather.data)
summary(weather.lmod.small)
summary(weather.lmod.all)
```

```
Call:
lm(formula = Temperature..C. ~ Wind.Speed..km.h. + Visibility..km. +
    Pressure..millibars., data = weather.data[1:30, ])

Residuals:
    Min      1Q  Median      3Q     Max
-6.7367 -1.4240 -0.3303  1.8014  6.0620

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          -36.76393  396.99271  -0.093  0.92693
Wind.Speed..km.h.      0.26616    0.11957   2.226  0.03490 *
Visibility..km.       -0.84184    0.26459  -3.182  0.00377 **
Pressure..millibars.   0.05542    0.38941   0.142  0.88793
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.931 on 26 degrees of freedom
Multiple R-squared:  0.4621,Adjusted R-squared:    0.4
F-statistic: 7.444 on 3 and 26 DF,  p-value: 0.0009344



Call:
lm(formula = Temperature..C. ~ Wind.Speed..km.h. + Visibility..km. +
    Pressure..millibars., data = weather.data)

Residuals:
     Min       1Q   Median       3Q      Max
-30.9163  -6.4825  -0.5336   6.0019  28.6052

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           6.2814847  0.3594689   17.47  < 2e-16 ***
Wind.Speed..km.h.    -0.0416818  0.0055575   -7.50 6.48e-14 ***
Visibility..km.       0.9640102  0.0100791   95.64  < 2e-16 ***
Pressure..millibars. -0.0035936  0.0003374  -10.65  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 9.025 on 54915 degrees of freedom
Multiple R-squared:  0.1444,Adjusted R-squared:  0.1444
F-statistic:  3090 on 3 and 54915 DF,  p-value: < 2.2e-16
```

We can confirm that the larger amount of data gets us better estimates.

[21]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2. (c) Interpreting Our Models (10 points) Answer the following questions and put your answer with the corresponding answer number.

1. TRUE/FALSE. The coefficient for `Pressure..millibars.` for the model fit to all the data is statistically signficant.
2. TRUE/FALSE. The coefficient for `Pressure..millibars.` for the model fit to a small amount of data is statistically significant.
3. What is the R^2 for the model fit to all of the data?
4. What is the R^2 for the model fit to a small amount of the data?
5. Which model explained more variablility in its respective dataset? Copy the correct model into this answer variable. Think about why this is the case!
6. TRUE/FALSE. Models fit to large amounts of data run the risk of having statistically significant coefficients, even if the predictor isn't practically significant to the response.

[22]:
```
prob.3.c.1 = NA

prob.3.c.2 = NA

prob.3.c.3 = NA

prob.3.c.4 = NA

# Save the selected model into this variable.
prob.3.c.5 = NA

prob.3.c.6 = NA

# your code here
prob.3.c.1 = TRUE

prob.3.c.2 = FALSE

prob.3.c.3 = 0.1444

prob.3.c.4 = 0.4621
```

```
prob.3.c.5 = mod.3

prob.3.c.6 = TRUE
```

[23]:
```
# TEST CELL
if (!test_that("Checking type() of answer", expect_is(prob.3.c.5, "lm"))){
    print("Make sure prob.3.c.5 is your selected linear model. Should be of␣
 ↪type 'lm'")
}
# This cell has hidden test cases that will run after submission.
```

[24]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[25]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[26]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[27]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[28]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```