Homework 6: ANOVA and Intervals

Mario Giacomazzo

October 04, 2023

Instructions:

The purpose of this homework assignment is to look more into understanding the ANOVA table and how intervals enhance our predictions. Make sure you read each question carefully. In each question, I will give you a task to do, and I will tell you what I want you to output. You can write as much code as you want in each code chunk, but make sure you complete the task and only print the output I asked you to print. Don't sort the data unless you are told to sort the data. You should remove the "#" sign in each code chunk before writing your code. Also, if you see the comment "#DO NOT CHANGE", then I don't want you to make any modifications to that code. **You should knit your RMD file to a PDF after you answer every question.**

After you are done, knit the RMarkdown file to PDF and submit the PDF to Gradescope under HW5.

Questions

Q1 (9 Points)

The table below is an ANOVA table from a simple linear regression fitted to a dataset with 60 observations. First knit the document to see what the output looks like in the Markdown table below:

Source	d.f.	Sum of Squares	Mean Square	F	P-value
Model	NA	NA	400	NA	NA
Residual	NA	500	NA		
Total	NA	NA			

I want you to fill-in the NA values of this ANOVA table based on the information I have given you in the table. You can put your work for finding the NA values in the code chunk below. **Carry your answers to 4 decimal places when performing calculations but round your final values in the table to two decimal places.**

You need to know that in R, `5e-8` is equivalent to scientific notation 5×10^{-8}.

Q2 (4 Points)

The dataset **Pines** comes from the **Stat2Data** package. You should start by looking at the help documentation to understand the contents of this dataset. After loading the package, `?Pines` can be useful to get descriptions of all the variables in the dataset. Run the code `?Pines` temporarily, then erase so that it is not present when you knit the document.

I want you to create a new dataframe called **Pines2** that only contains the two variables named *Hgt96* and *Diam97* and only contains observations that have a non-missing measurements for these two variables.

Considering using the `na.omit()` function which removes all observations where there is at least one missing value in one of the columns.

Use the `str()` function to print out a preview of **Pines2** and use the `plot()` function to visualize the relationship of *Diam97* versus *Hgt96*. The output from these two functions should be your only output.

```
data("Pines") #DO NOT CHANGE
```

Q3 (2 Points)

Add variables to **Pines2** named *sqH96* and *sqD97* that are square roots of the original variables *Hgt96* and *Diam97*, respectively. Then, create the same scatterplot from the previous question with the square roots of the variables replacing the original variables.

Q4 (2 Points)

There is one data point that has high leverage and is clearly outside the pattern of the rest of the data more than all the other points. I want you to identify this point and create a data frame called **Pines3** that contains all the data from **Pines2** except for this problematic point which we are assuming is an error/mistake.

The `which.min()` and `which.max()` functions can be useful for identifying the location of a minimum or maximum in a vector. The first two lines of code show how these functions work. Using one of these functions may be more useful then scrolling through the data to find the exact row where the problem exists.

Also, when we are subsetting using the bracket notation, you can go from selecting observation(s) to removing observation(s) by converting positive integers to negative integers. The next two lines of code illustrate how this works on a vector.

I would learn how these two ideas can be combined to identify the exact row that contains the bad observation. Use the `str()` function on **Pines3** to confirm that this observation is removed.

```
which.min(c(4,9,5,3,2,5,6,2,3,1,8)) #DO NOT CHANGE
```

```
## [1] 10
```

```
which.max(c(4,9,5,3,2,5,6,2,3,1,8)) #DO NOT CHANGE
```

```
## [1] 2
```

```
c(3,4,5,6,7,8)[c(1,3)]  #DO NOT CHANGE
```

```
## [1] 3 5
```

```
c(3,4,5,6,7,8)[-c(1,3)]  #DO NOT CHANGE
```

```
## [1] 4 6 7 8
```

Q5 (6 Points)

I want you to fit a linear model for the relationship *sqD97* vs *sqH96* based on the data in **Pines3**. I want you to print out the ANOVA table from this model and I want you to calculate the standard error of the regression and the r-squared statistic only using the numbers in this table. In your output, I should see the ANOVA table and the final calculations of the two statistics mentioned above based on the numbers in that table.

After this, round your numerical values for the standard error and r-squared to two decimal places, and write them in the appropriate space below the code chunk. Don't round until you get your final answer.

Standard Error of Regression: REPLACE ALL CAPS WITH YOUR ANSWER

r-squared: REPLACE ALL CAPS WITH YOUR ANSWER

Q6 (2 Points)

Perform the appropriate t-test for the correlation between *sqD97* and *sqH96*. Also, use the `summary()` function on your model. Compare the p-values from the t-test for the slope and the t-test for the correlation. Notice they are not identically presented, but they are identical. This is just due to different rounding rules used in the two outputs.

I only want to see the output from the correlation test and the `summary()` function.

Q7 (4 Points)

Suppose there were two Pine trees that were not in the dataset **Pines3**. One of these, pine trees had a height of 300cm and the other had a height of 410cm in 1996. I want to generate two intervals for the purpose of predicting the actual tree trunk diameter in centimeters for each of these trees in 1997.

To do this follow these steps:

1. Create a data.frame called **newx** that only contains the tree heights for these two trees in 1996. Use the same original variable name from **Pines**.
2. Create a variable of the same name of a variable used before that is the square root of these heights. Add this variable to **newx**.
3. Use the `predict()` function appropriately to apply the fitted model to the data in **newx**. Make sure you obtain the appropriate interval for this situation. Save the output from the `predict()` function into an object called **predictx**.
4. Remember that are model predicts the square root of the diameter and not the actual diameter. Therefore, run the code `predictx^2` to untransform all the numbers in the table back to their original units. This should be the only output from this code.

This table shows the predicted diameters in cm for these two trees in 1997 along with the appropriate prediction intervals. Look at the original scatter plot and think about if your final values make sense. Read the textbook about how to appropriately interpret and use these prediction intervals for both of these trees.