Lab 3: Relationship between global cesarean delivery rates and GDP

Felicia Liu

09/10/2020

Instructions

- Due date: Friday, September 11 at 11:59pm PST.
- Late penalty: 50% late penalty if submitted within 24 hours of due date, no marks for assignments submitted thereafter.
- This assignment is graded on **correct completion**, all or nothing. You must pass all public tests and submit the assignment for credit.
- Submission process: Follow the submission instructions on the final page. Make sure you do not remove any `\newpage` tags or rename this file, as this will break the submission.

Start by loading the required libraries, reading in the data and adding on a variable:

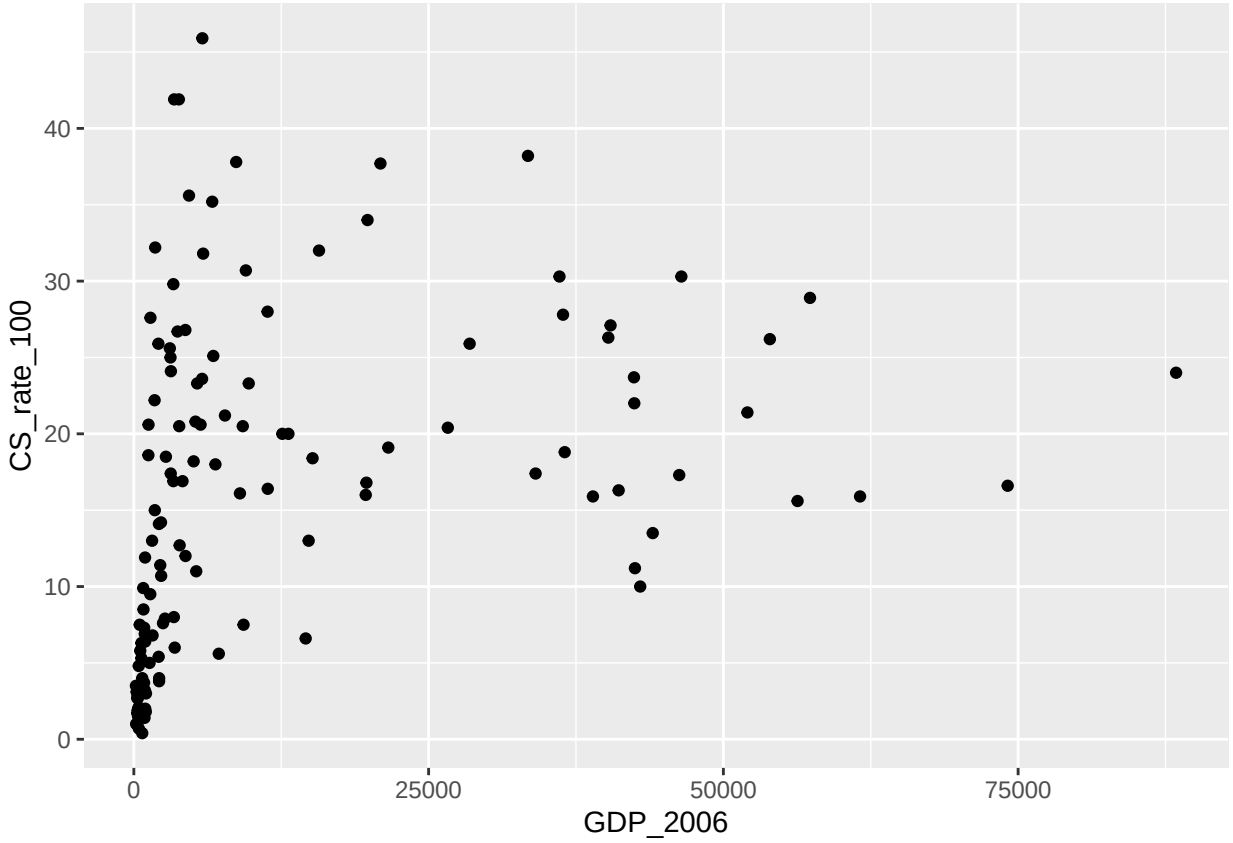
```
library(dplyr)
library(ggplot2)
library(readr)
library(broom)

CS_data <- read_csv("cesarean.csv")
```

```
## Parsed with column specification:
## cols(
##   Country_Name = col_character(),
##   CountryCode = col_character(),
##   Births_Per_1000 = col_double(),
##   Income_Group = col_character(),
##   Region = col_character(),
##   GDP_2006 = col_double(),
##   CS_rate = col_double()
## )
```

```
# This code re-orders the variable Income_Group in the specified order.
# Note that it *does not* change the order of the data frame (like arrange() does)
# Rather, it specifies the order the data will be plotted.
# This will make more sense when we plot the data using Income_Group, and then
# again using Income_Group_order
CS_data$Income_Group <- forcats::fct_relevel(CS_data$Income_Group,
                                             "Low income", "Lower middle income",
                                             "Upper middle income", "High income: nonOECD",
                                             "High income: OECD")

CS_data <- CS_data %>% mutate(CS_rate_100 = CS_rate*100)
```

1. [1 point] Make a scatter plot between `CS_rate_100` and `GDP_2006`:

```
p1 <- ggplot(CS_data, aes(x = GDP_2006, y = CS_rate_100)) + geom_point()
p1
```




```
check_problem1()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: You are using the correct dataset!"
## [1] "Checkpoint 3 Passed: Correct variable plotted!"
## [1] "Checkpoint 4 Passed: Correct variable plotted!"
## [1] "Checkpoint 5 Passed: You defined a scatterplot!"
##
## Problem 1
## Checkpoints Passed: 5
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

In your plot, you might notice that many of the points are condensed towards the lower left corner. And you might recall from the lab and assignment that the distributions of both cesarean delivery rate and GDP covered a wide range of values. Both of these variables are good candidates for log transformations to spread out the range of data at the lowest levels.

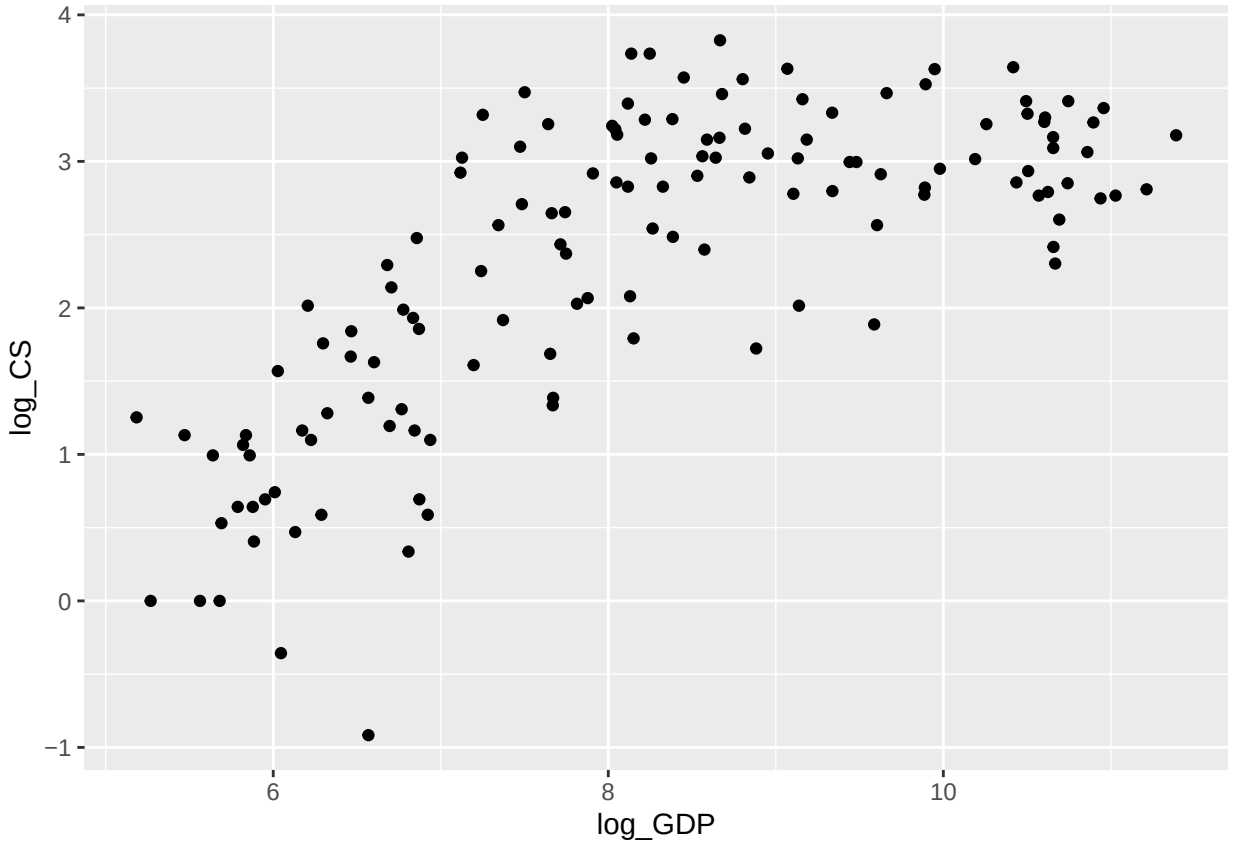
2.[1 point] Using the `mutate()` function, add two new logged variables to the data set `CS_data` and assign this new data set to `CS_data_log`. Call the variables `log_CS` and `log_GDP`. Use base e, also known as natural logarithms, to create the logged variables:

```
CS_data_log <- CS_data %>% mutate(log_CS = log(CS_rate_100), log_GDP = log(GDP_2006))


check_problem2()
```

```
## [1] "Checkpoint 1 Passed: You correctly named you dataset!"
## [1] "Checkpoint 2 Passed: You correctly transformed CS_rate_100!"
## [1] "Checkpoint 3 Passed: You correctly transformed GDP_2006!"
##
## Problem 2
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

3. [1 point] Remake the scatter plot using the logged variables:

```
p3 <- ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) + geom_point()
p3
```



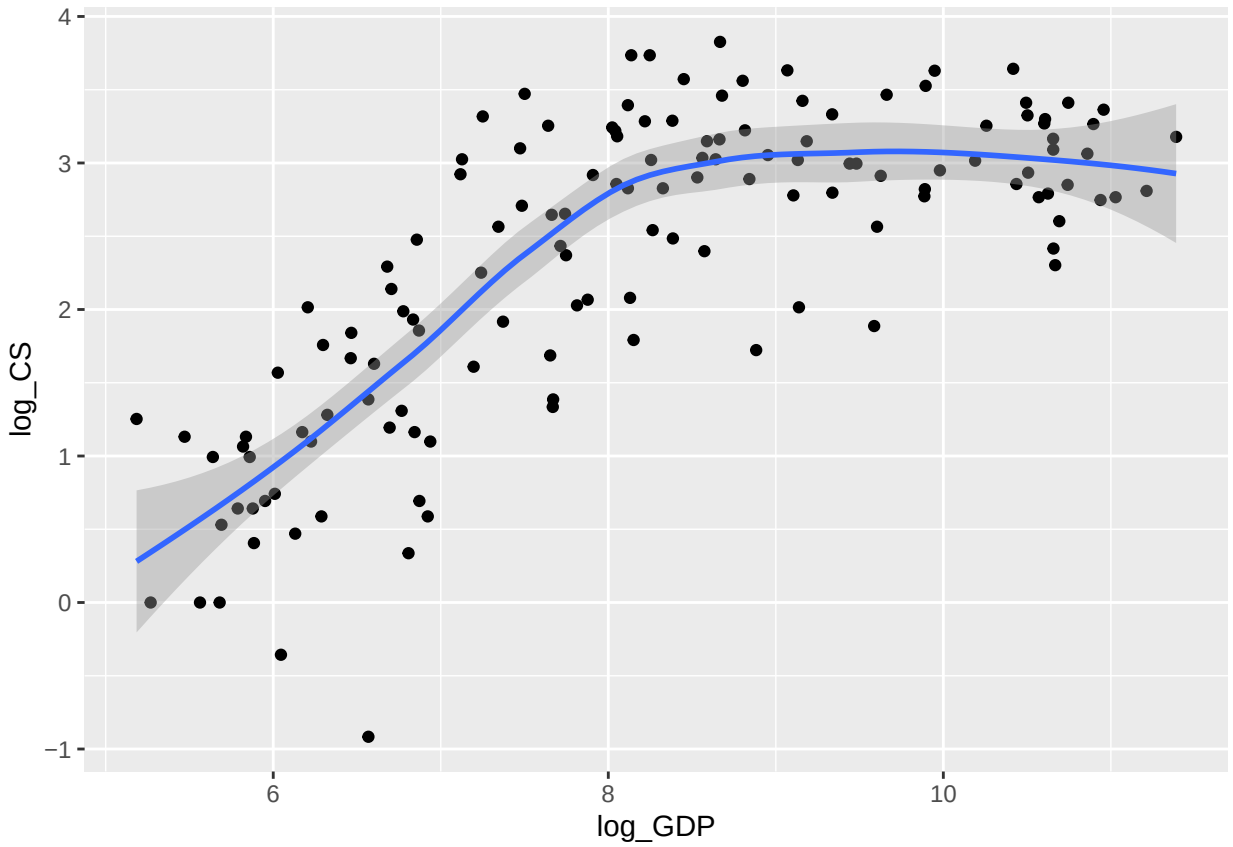

```
check_problem3()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: You've used the right dataset!"
## [1] "Checkpoint 3 Passed: You plotted the right variable!"
## [1] "Checkpoint 4 Passed: You've plotted the right variable!"
## [1] "Checkpoint 5 Passed: You correctly defined a scatter ggplot!"
##
## Problem 3
## Checkpoints Passed: 5
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

4. [1 point] A geom that you have not yet learnt is geom_smooth(). This geom can fit a curve to the data. Extend your ggplot() code by adding geom_smooth() to it:

```
p4 <- ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) + geom_point() + geom_smooth()
p4
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
check_problem4()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: You used the correct dataset!"
## [1] "Checkpoint 3 Passed: You plotted the correct variable!"
## [1] "Checkpoint 4 Passed: You plotted the correct variable!"
## [1] "Checkpoint 5 Passed: You've defined a scatterplot in ggplot!"
## [1] "Checkpoint 6 Passed: You've defined a geom_smooth in ggplot!"
##
## Problem 4
## Checkpoints Passed: 6
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

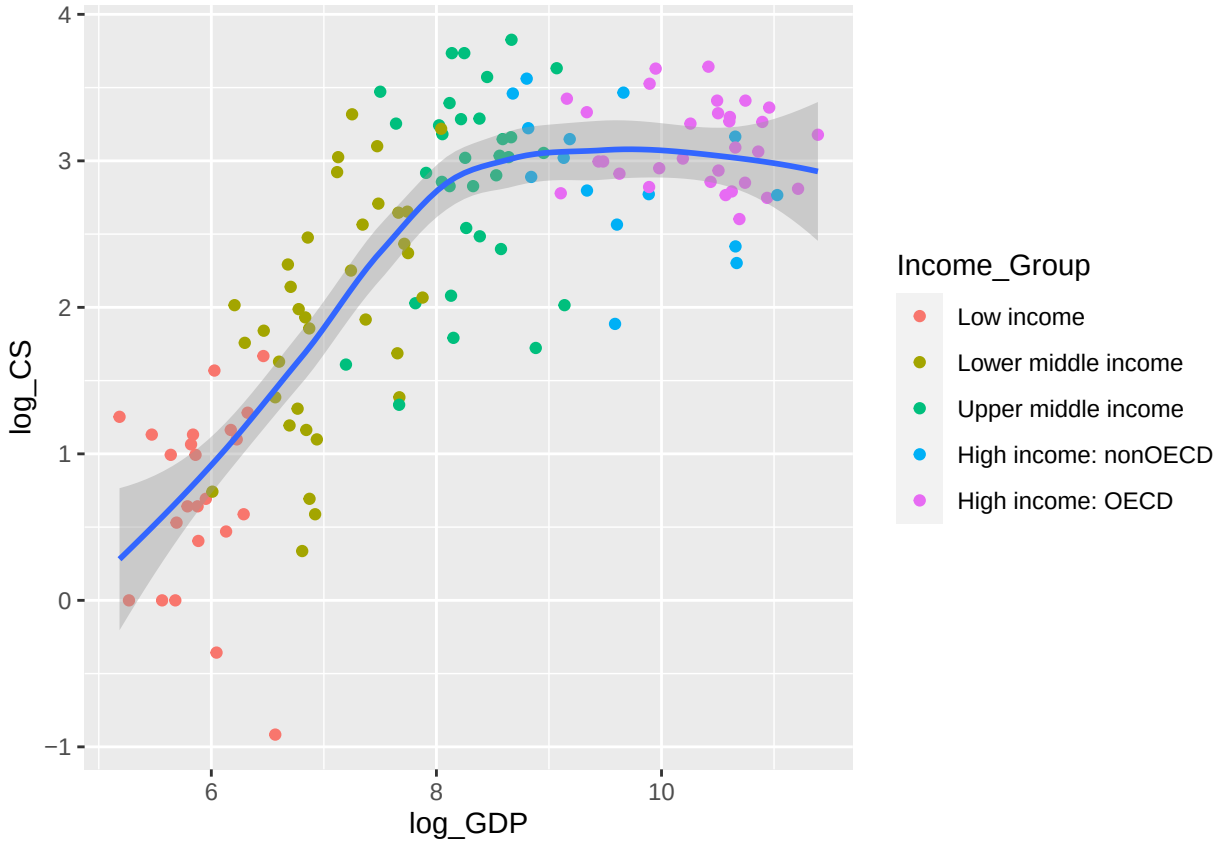
Does the relationship between logged GDP and logged CS look linear?

No, it does not look linear. It looks curved.

5. [1 point] Modify your scatter plot by linking the color of the points to the variable `Income_Group`.

```
p5 <- ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) +
  geom_point(aes(col = Income_Group)) + geom_smooth()
p5
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
check_problem5()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: You used the correct dataset!"
## [1] "Checkpoint 3 Passed: You plotted the correct variable!"
## [1] "Checkpoint 4 Passed: You plotted the correct variable!"
## [1] "Checkpoint 5 Passed: You've defined a scatterplot!"
## [1] "Checkpoint 6 Passed: You've defined a geom_smooth in ggplot!"
## [1] "Checkpoint 7 Passed: You've set the plot to color by Income_Group!"
##
## Problem 5
## Checkpoints Passed: 7
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Based on this colored scatter plot, it looks like the relationship is linear for those countries that are not categorized as one of the two high income categories.

6. [1 point] For this lab, we would like to use linear regression. To do this, use a `dplyr` function to make a new data set called `CS_data_sub` that only contains the low-, lower-middle, and upper-middle income countries (hint: You might want to look at the data to see exactly what these levels are called in the data set):

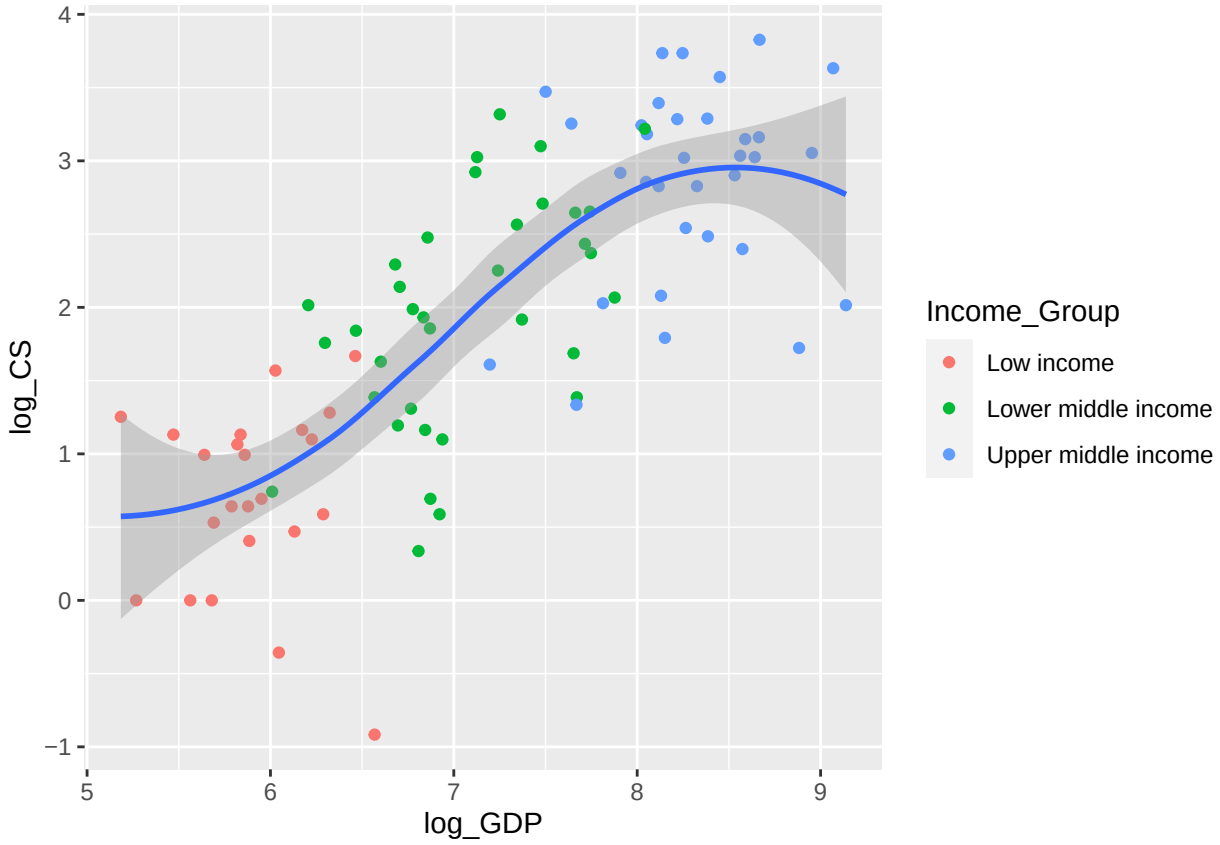
```
CS_data_sub <- CS_data_log %>% filter(Income_Group %in% c("Low income",
            "Lower middle income", "Upper middle income"))


check_problem6()
```

```
## [1] "Checkpoint 1 Passed: You've named your new dataset correctly!"
## [1] "Checkpoint 2 Passed: You've filtered Low income Group correctly!"
## [1] "Checkpoint 3 Passed: You've filtered Lower middle income Group correctly!"
## [1] "Checkpoint 4 Passed: You've filtered Upper middle income Group correctly!"
## [1] "Checkpoint 5 Passed: You've excluded High income: nonOECD group correctly!"
## [1] "Checkpoint 6 Passed: You've excluded High income: OECD group correctly!"
##
## Problem 6
## Checkpoints Passed: 6
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

7. [1 point] Remake the last scatter plot, this time using `CS_data_sub` to see if the relationship looks approximately linear between the logged variables:

```
p7 <- ggplot(CS_data_sub, aes(x = log_GDP, y = log_CS)) +
  geom_point(aes(col = Income_Group)) + geom_smooth()
p7
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
check_problem7()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: You've used the right dataset!"
## [1] "Checkpoint 3 Passed: You've plotted the right variable!"
## [1] "Checkpoint 4 Passed: You've plotted the right variable!"
## [1] "Checkpoint 5 Passed: You've defined a scatterplot in ggplot!"
## [1] "Checkpoint 6 Passed: You've defined a geom_smooth in ggplot!"
## [1] "Checkpoint 7 Passed: You've set the plot to color by Income_Group!"
##
## Problem 7
## Checkpoints Passed: 7
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

8. [1 point] Given that the relationship is approximately linear, use linear regression to model the relationship between `log_CS` as the response variable and `log_GDP` as the explanatory variable. Don't forget to specify the correct data set!:

```r
p8 <- lm(log_CS ~ log_GDP, data = CS_data_sub)
p8
```

```
##
## Call:
## lm(formula = log_CS ~ log_GDP, data = CS_data_sub)
##
## Coefficients:
## (Intercept)      log_GDP
##     -3.9405       0.8193
```

```r
check_problem8()
```

```
## [1] "Checkpoint 1 Passed: You've chosen the correct variable for the model!"
## [1] "Checkpoint 2 Passed: You've chosen the correct variable for the model!"
##
## Problem 8
## Checkpoints Passed: 2
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

9.Interpret the slope estimate:

The slope estimate is 0.8193, so a unit increase in the natural log of 2006 GDP would lead to 0.8193 unit increase in the natural log of cesarean delivery rates.

10.Estimate what the cesarean delivery rate would be for a country with a GDP of 2000. Outline the steps you take to calculate your answer and provide an interpretation. Round your final answer to one decimal place.

First, we would find the equation of the linear model. Using the standard form of y=mx+b, we could come up with the equation y=0.8193x-3.9405. The y would be the natural log of the estimated cesarean rate, since everything is in log. Second, we plug log(2000) into x and will get -1.235 for y. Because y is the natural log of the cesarean rate we want to find, we have -1.235 = log(z), with z being the estimated cesarean rate. To find z, we take eˆ(-1.236) and get 0.29055. So for a country with an 2006 GDP of 2000, we would predict the cesarean delivery rate to be approximately 0.29055.

11. Is it appropriate to use the model to predict the cesarean delivery rate for a country with a GDP of 50,000? Why or why not? Based on the relationship in the full data set, would you expect the linear model to over or under predict?

[TODO: YOUR ANSWER HERE]

Check your score

Click on the middle icon on the top right of this code chunk (with the downwards gray arrow and green bar) to run all your code in order. Then, run this chunk to check your score.

```
# Just run this chunk.
total_score()
```

```
##              Test Points_Possible       Type
## Problem 1  PASSED                1 autograded
## Problem 2  PASSED                1 autograded
## Problem 3  PASSED                1 autograded
## Problem 4  PASSED                1 autograded
## Problem 5  PASSED                1 autograded
## Problem 6  PASSED                1 autograded
## Problem 7  PASSED                1 autograded
## Problem 8  PASSED                1 autograded
## Problem 9  FAILED                0 autograded
## Problem 10 FAILED                0 autograded
## Problem 11 FAILED                0 autograded
```

Submission

For assignments in this class, you'll be submitting using the **Terminal** tab in the pane below. In order for the submission to work properly, make sure that:

1. Any image files you add that are needed to knit the file are in the `src` folder and file paths are specified accordingly.
2. You **have not changed the file name** of the assignment.
3. The file knits properly.

Once you have checked these items, you can proceed to submit your assignment.

1. Click on the **Terminal** tab in the pane below.
2. Copy-paste the following line of code into the terminal and press enter.

cd; cd ph142-fa20/lab/lab03; python3 turn_in.py

3. Follow the prompts to enter your Gradescope username and password.
4. If the submission is successful, you should see "Submission successful!" appear as output.
5. If the submission fails, try to diagnose the issue using the error messages–if you have problems, post on Piazza under the post "Submission Issues".

The late policy will be strictly enforced, **no matter the reason**, including submission issues, so be sure to submit early enough to have time to diagnose issues if problems arise.