Assignment 2: Summarize global cesarean delivery rates and GDP across 137 countries

Felicia Liu

09/04/2020

- Solutions will be released on Tuesday, September 8.
- This semester, homework assignments are for practice only and will not be turned in for marks.

Helpful hints:

- Every function you need to use was taught during lecture! So you may need to revisit the lecture code to help you along by opening the relevant files on Datahub. Alternatively, you may wish to view the code in the condensed PDFs posted on the course website. Good luck!
- Knit your file early and often to minimize knitting errors! If you copy and paste code for the slides, you are bound to get an error that is hard to diagnose. Typing out the code is the way to smooth knitting! We recommend knitting your file each time after you write a few sentences/add a new code chunk, so you can detect the source of knitting errors more easily. This will save you and the GSIs from frustration! **You must knit correctly before submitting.**
- If your code runs off the page of the knitted PDF then you will LOSE POINTS! To avoid this, have a look at your knitted PDF and ensure all the code fits in the file (you can easily view it on Gradescope via the provided link after submitting). If it doesn't look right, go back to your .Rmd file and add spaces (new lines) using the return or enter key so that the code runs onto the next line.

Summarizing global cesarean delivery rates and GDP across 137 countries

Introduction

Recall from this week's lab that we constructed bar charts and histograms to explore a data set that looked at global rates of cesarean delivery and GDP. If you need a refresher, you can view your knitted file from lab and remind yourself what you found.

In this week's assignment, you will describe these distributions using numbers. You will investigate the **mean** and **median** of the distribution of GDP. You will also examine the distribution of cesarean delivery separately for countries of varying income levels. Lastly, you will describe the **spread** of the distributions using **quartiles** and make a **box plot**.

Execute this code chunk to load the required libraries:

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

Just like last time, read in the data that is stored as a .csv file and assign it to an object called `CS_data`. We will also use dplyr's `mutate()` to create the new cesarean delivery variable that ranges between 0 and 100:

```
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 reorders the Factor variable `Income_Group` in the
# order specified in this function. This will affect the order the ggplot
# panels are shown in question 8 when we use `facet_wrap()`.
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. [1.5 points] Fill in the blanks indicated by "—-" by saving the answer to each blank in the code chunk below. Make sure you capitalize correctly, as R is case-sensitive.

The function `mutate()` takes the old variable called –**(a)**– and multiplies it by — to make a new variable called –**(b)**–.

```
a <- "CS_rate"
b <- "CS_rate_100"

# ------- REMINDER -------
# The checks on this homework are only provided as sanity checks.
# They do not guarantee correctness.
# ------------------------
check_problem1()
```

```
## [1] "Checkpoint 1 Passed: You got the a correct!"
## [1] "Checkpoint 2 Passed: You got the b correct!"
##
## Problem 1
## Checkpoints Passed: 2
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Investigate what would have happened had we not assigned the data using `<-` to `CS_data`? Re-run the code without the assignment and see what happens.

```r
# First, let's re-read in the data as we did in the previous chunk
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()
## )
```

```r
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")


# Now, we try again without the re-assignment to CS_data
CS_data %>% mutate(CS_rate_100 = CS_rate*100)
```

```
## # A tibble: 137 x 8
##    Country_Name CountryCode Births_Per_1000 Income_Group Region GDP_2006
##    <chr>        <chr>                 <dbl> <fct>        <chr>     <dbl>
##  1 Albania      ALB                      46 Upper middl~ Europ~    3052.
##  2 Andorra      AND                       1 High income~ Europ~   42417.
##  3 United Arab~ ARE                      63 High income~ Middl~   42950.
##  4 Argentina    ARG                     689 High income~ Latin~    6649.
##  5 Armenia      ARM                      47 Lower middl~ Europ~    2127.
##  6 Australia    AUS                     267 High income~ East ~   36101.
##  7 Austria      AUT                      76 High income~ Europ~   40431.
##  8 Azerbaijan   AZE                     166 Upper middl~ Europ~    2473.
##  9 Belgium      BEL                     119 High income~ Europ~   38936.
## 10 Benin        BEN                     342 Low income   Sub-S~     557.
## # ... with 127 more rows, and 2 more variables: CS_rate <dbl>,
## #   CS_rate_100 <dbl>
```

```r
# check the variables on CS_data
names(CS_data)
```

```
## [1] "Country_Name"    "CountryCode"     "Births_Per_1000" "Income_Group"
## [5] "Region"          "GDP_2006"        "CS_rate"
```

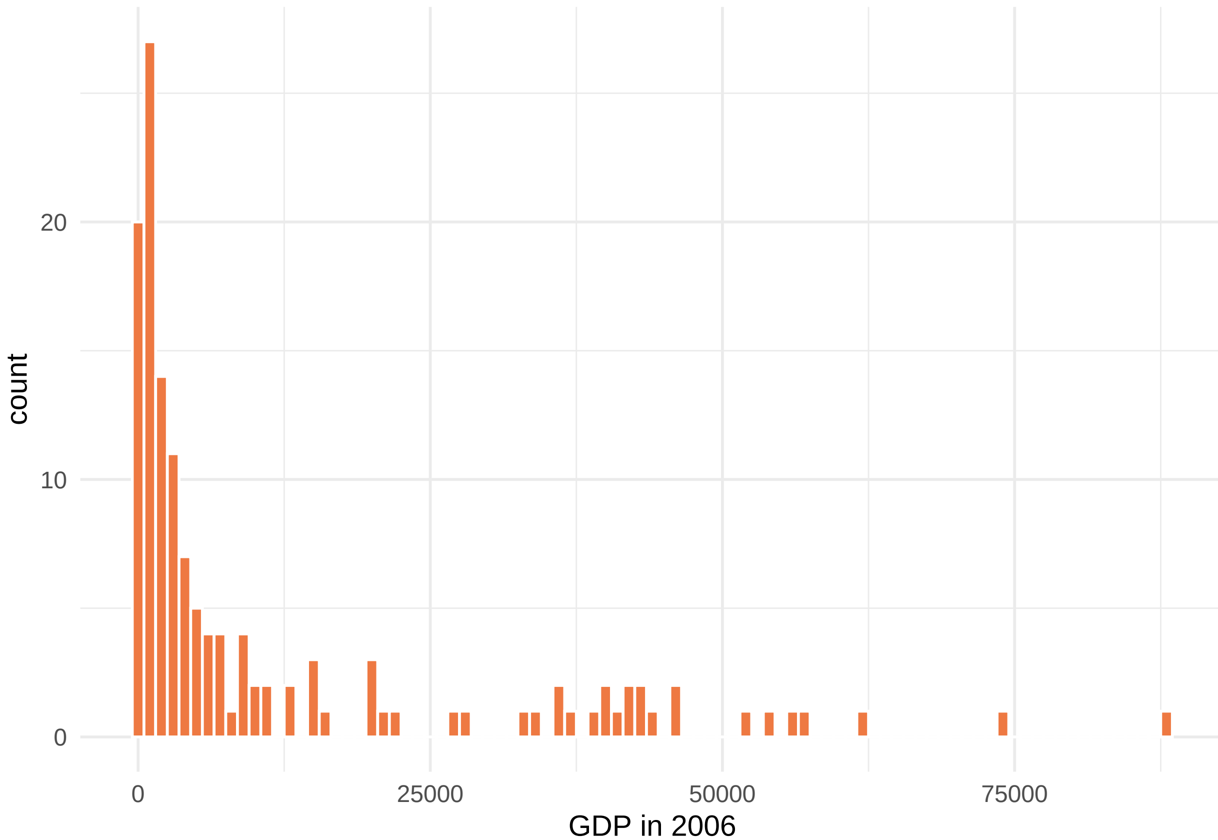
Did `CS_rate_100` get added to the data set? No. You can tell by using `head(CS_data)` to view the first few rows and notice that the variable hasn't been added. This is because when we don't assign the output to anything, it just prints it out for us to see. Nothing is saved. So, we want to save the output by assigning the result of the code to a variable, which in this case, we used `CS_data`. In general, you want to use **new variable names** at every significant step in your analysis as you work with your data, so that you have access to the data at all those significant stages. However, if you are performing multiple small operations on the same dataset, you can overwrite the original variable, since you know you won't be needing the in-between steps anyway.

```r
# This overwrites the original CS_data object
CS_data <- CS_data %>% mutate(CS_rate_100 = CS_rate*100)
```

GDP: Summarizing measures of centrality

Recall your histogram of GDP from this week's lab:

```
ggplot(data = CS_data, aes(x = GDP_2006)) +
  geom_histogram(col = "white", fill = "sienna2", binwidth = 1000) +
  xlab("GDP in 2006") +
  theme_minimal()
```



2. [1 point] Describe the shape of this distribution. Is it "skewed left", "skewed right", "symmetric", or "bimodal"? Uncomment one of the possible choices.

```
# p2 <- "skewed left"
p2 <- "skewed right"
# p2 <- "symmetric"
# p2 <- "bimodal"

check_problem2()
```

```
## [1] "Checkpoint 1 Passed: You chose the correct shape of distribution!"
##
## Problem 2
## Checkpoints Passed: 1
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

3. [1 point] Based on your answer, will the mean be approximately the "same", "larger than", or "smaller than" the median?

```r
# p3 <- "same"
p3 <- "larger than"
# p3 <- "smaller than"

# This only checks that you've selected an answer, not its correctness.
check_problem3()
```

```
## [1] "Checkpoint 1 Passed: Correct choice!"
##
## Problem 3
## Checkpoints Passed: 1
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

4. [3 points] Describe, in words, how the mean and median are calculated:

The mean is calculated by adding up all the numbers and dividing that sum by how many numbers there are. The median is calculated by ordering a list of numbers from smallest to largest, and then finding the number at the very middle or the list.

To calculate the mean and median in R, we can use the `summarize()` function from the `dplyr` package. The `summarize()` function is used anytime we want to take a variable and summarize something about it into one number, like it's mean or median. Here is the code to summarize `GDP_2006`'s mean and print it out to the screen. In the code, we name the mean `mean_GDP` and output the result to the screen:

```
GDP_summary <- CS_data %>% summarize(mean_GDP = mean(GDP_2006))
GDP_summary
```

```
## # A tibble: 1 x 1
##   mean_GDP
##      <dbl>
## 1   11791.
```

5. [1 point] Extend the above code to also summarize the median. Call the median summary `median_GDP`. Assign this summary to `GDP_summary` (it will overwrite the previous version):

```
GDP_summary <- CS_data %>% summarize(mean_GDP = mean(GDP_2006), median_GDP = median(GDP_2006))
GDP_summary
```

```
## # A tibble: 1 x 2
##   mean_GDP median_GDP
##      <dbl>      <dbl>
## 1   11791.      3351.
```

```
check_problem5()
```

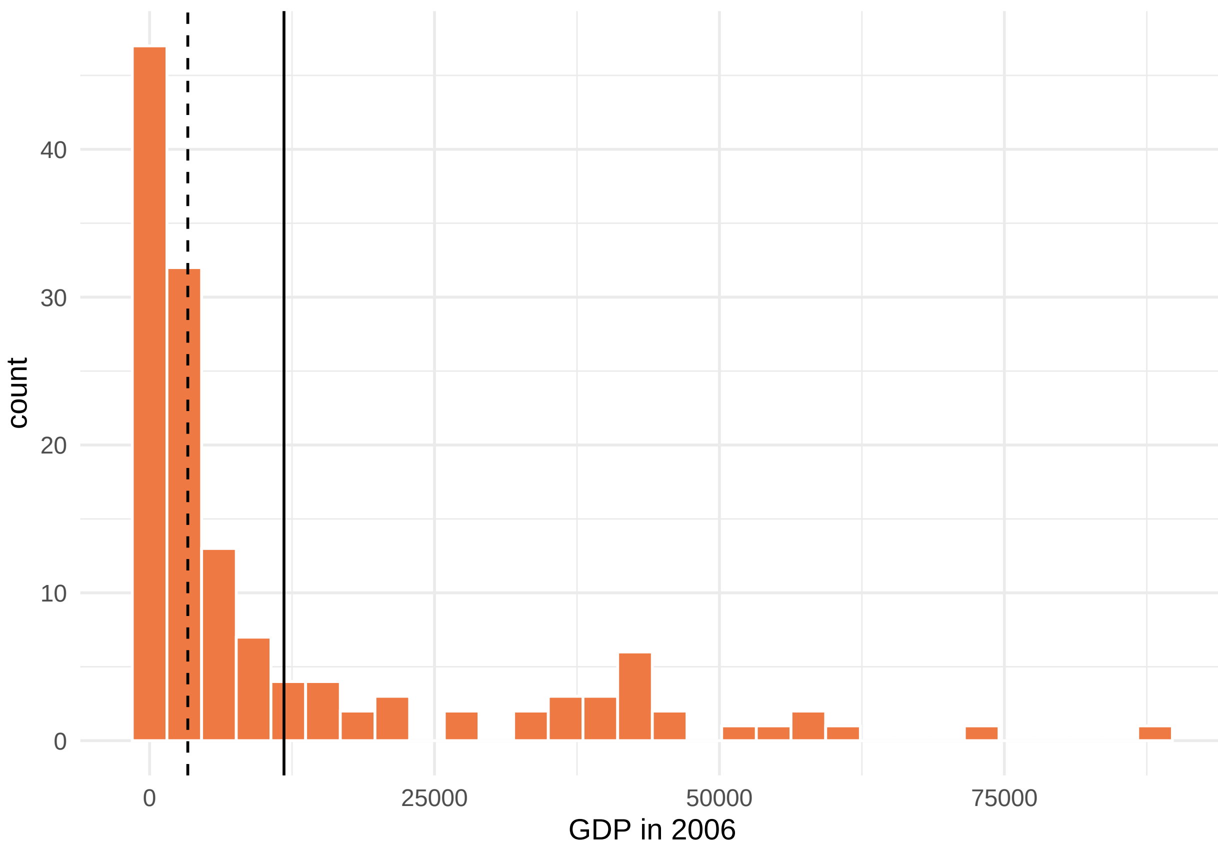
```
## [1] "Checkpoint 1 Passed: Correct mean value has been calculated!"
## [1] "Checkpoint 2 Passed: Correct median value has been calculated!"
##
## Problem 5
## Checkpoints Passed: 2
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

6. [2 points] `geom_vline()` **can be used to add the mean and the median to the histogram shown above. This** `geom_vline()` **adds a vertical line to the graph. You need to specify where to add the line by passing it an "x-intercept" argument. Remove the comments (i.e., the three "#") from the code below and update the** `geom_vline()` **code to plot lines at the mean and median by telling it the mean and median estimates. The argument** `lty=1` **(standing for line type) will plot a solid line and** `lty=2` **will plot a dashed line.**

For the purposes of this question, please assign xintercept to a plain NUMERIC, not a variable or expression

```
p6 <- ggplot(data = CS_data, aes(x = GDP_2006)) +
        geom_histogram(col = "white", fill = "sienna2") +
        xlab("GDP in 2006") +
        theme_minimal() +
        geom_vline(aes(xintercept = 11791), lty=1) +
        geom_vline(aes(xintercept = 3351), lty=2)
p6
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# REMINDER: this is only a sanity check; it doesn't check for accuracy
check_problem6()
```

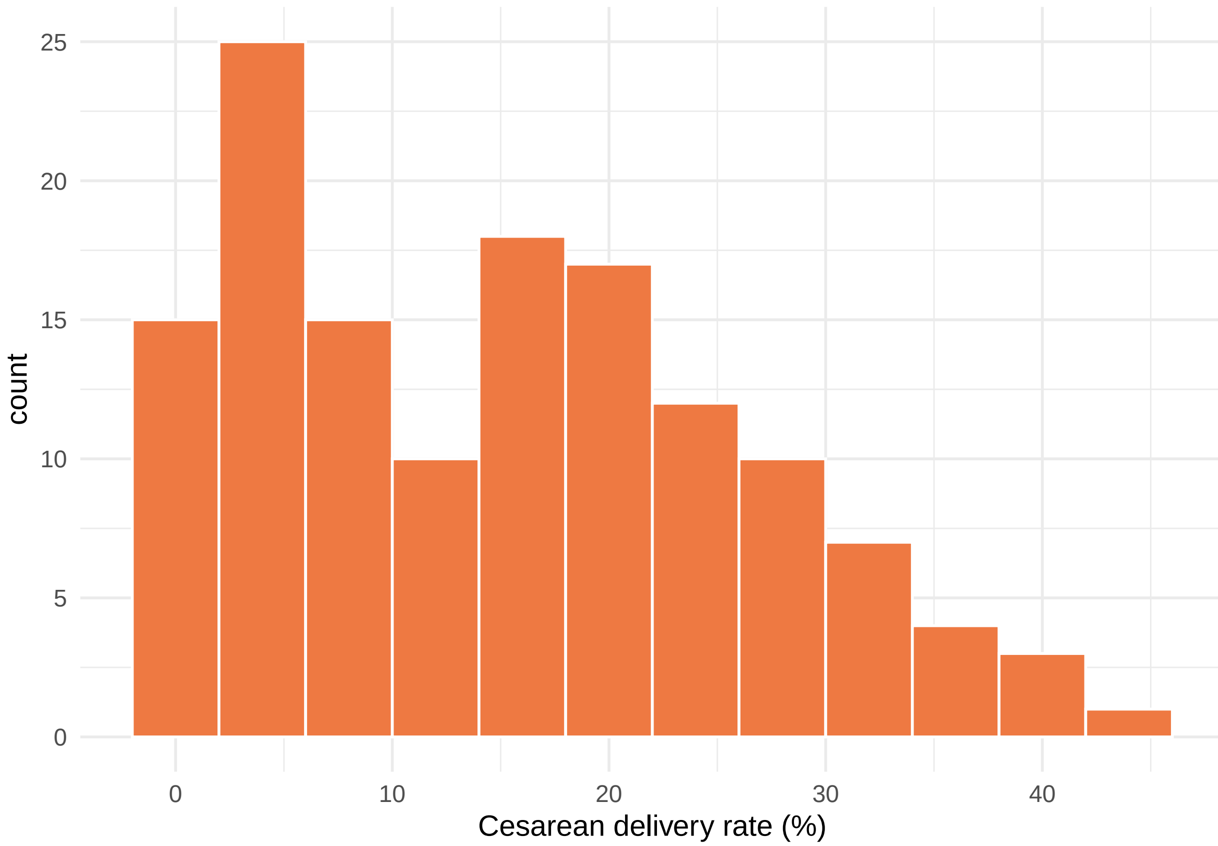
```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: A vertical line of the correct mean added!"
## [1] "Checkpoint 3 Passed: A vertical line of the correct median added!"
##
## Problem 6
```

```
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Summarizing the distribution of cesarean delivery

Recall the distribution of cesarean delivery rates across countries:

```
ggplot(data = CS_data, aes(x = CS_rate_100)) +
  geom_histogram(binwidth = 4, col = "white", fill = "sienna2") +
  xlab("Cesarean delivery rate (%)") +
  theme_minimal()
```



7. [1 point] Describe the shape of this distribution. Is it "skewed left", "skewed right", "symmetric", or "bimodal"?

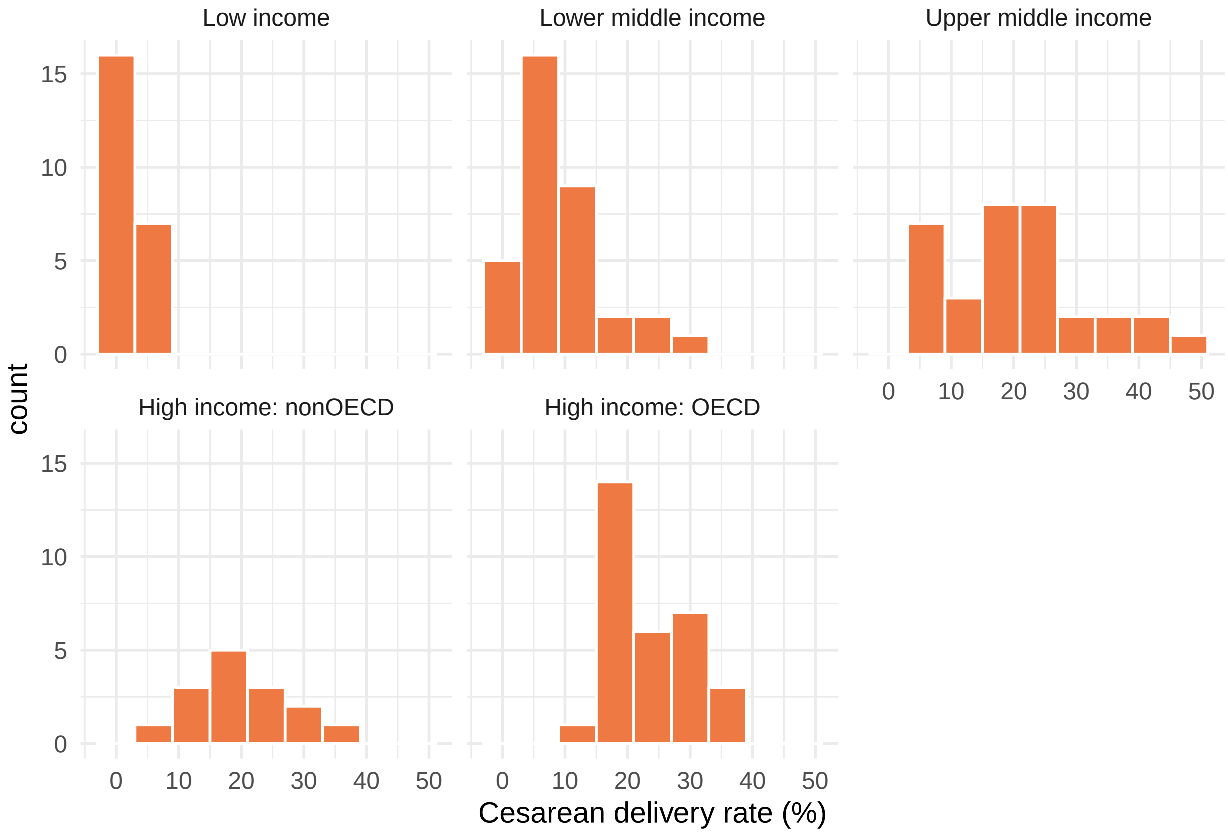
```r
# p7 <- "skewed left"
# p7 <- "skewed right"
# p7 <- "symmetric"
p7 <- "bimodal"

# This only checks that you've selected an answer, not its correctness.
check_problem7()
```

```
## [1] "Checkpoint 1 Passed: You chose the correct shape of distribution!"
##
## Problem 7
## Checkpoints Passed: 1
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

There appears to be multiple peaks which sometimes points to there being another variable that might explain the peaks. We can make a separate histogram for each income group using the `facet_wrap()` function.

8. [1 point] Extend the ggplot code given below using the `facet_wrap()` statement to make a separate histogram for each level of the `Income_Group` variable:

```
p8 <- ggplot(data = CS_data, aes(x = CS_rate_100)) +
        geom_histogram(binwidth = 6, col = "white", fill = "sienna2") +
        xlab("Cesarean delivery rate (%)") +
        theme_minimal() + facet_wrap(~ Income_Group)
p8
```




```
check_problem8()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: A separate histogram for each level of the `Income_Group` variable has bee
##
## Problem 8
## Checkpoints Passed: 2
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

9. [2 points] Based on this plot and the previous one, describe why the data had two peaks

The ceserean delivery rate peaks of lower income versus higher income countries are different, and therefore create two peaks in the combined data histogram.

10. [1 point] Why might lower income countries have lower rates of cesarean delivery?

Lower income countries might not have the proper medical equipment to carry out ceserean deliveries. These countries could be struggling financially so ceserean deliveries aren't as available as in higher income countries.

11. [2 points] Calculate the `mean_CS` and `median_CS` of `CS_rate_100` using only one `summarize()` command. Assign this summary to the name `CS_summary` and then print the results by typing `CS_summary` so you can see the contents.

```
CS_summary <- CS_data %>% summarize(mean_CS = mean(CS_rate_100), median_CS = median(CS_rate_100))
CS_summary
```

```
## # A tibble: 1 x 2
##   mean_CS median_CS
##     <dbl>     <dbl>
## 1    15.3      15.6
```

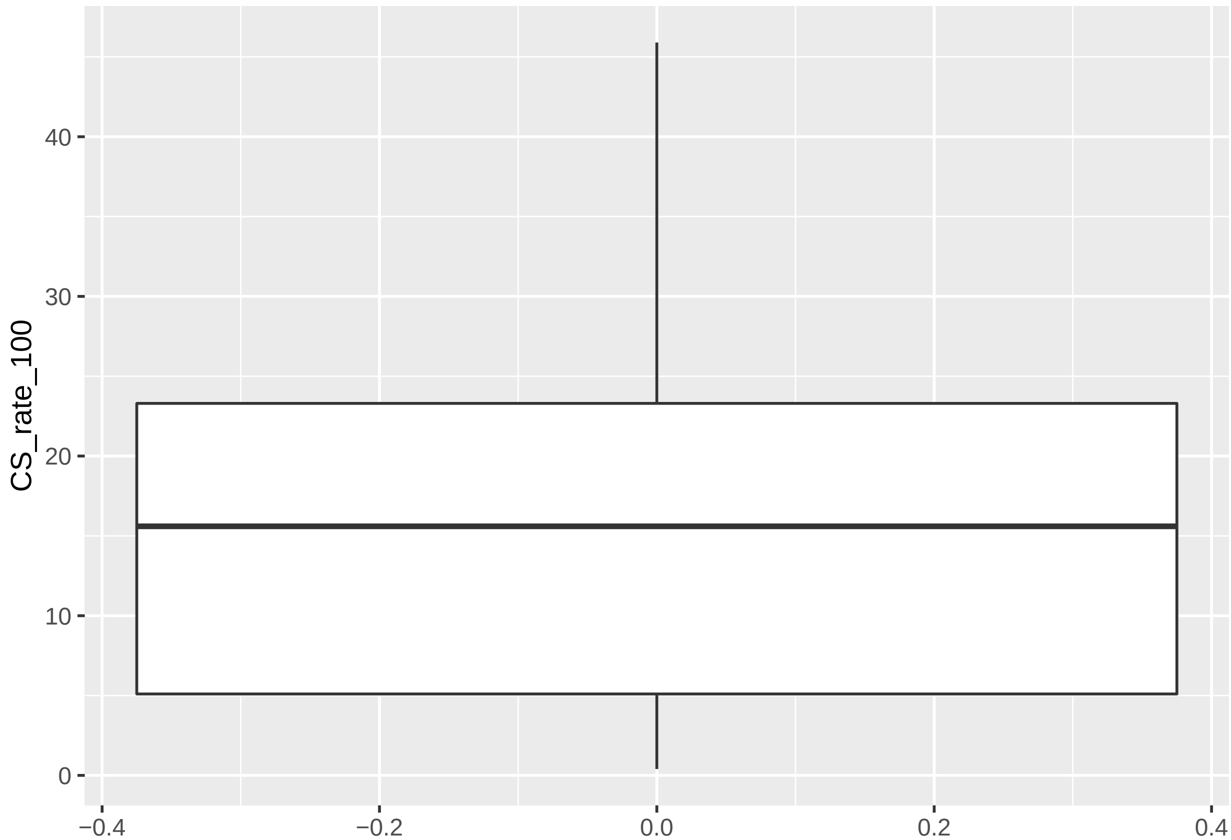
```
check_problem11()
```

```
## [1] "Checkpoint 1 Passed: Correct mean value has been calculated!"
## [1] "Checkpoint 2 Passed: Correct median value has been calculated!"
##
## Problem 11
## Checkpoints Passed: 2
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Measures of variation

12. [2 marks] Use ggplot2 to make a boxplot of the distribution of `CS_rate_100`

```
p12 <- ggplot(CS_data, aes(y = CS_rate_100)) + geom_boxplot()
p12
```




```
check_problem12()
```

```
## [1] "Checkpoint 1 Passed: A ggplot has been defined"
## [1] "Checkpoint 2 Passed: The correct y axis has been defined!"
## [1] "Checkpoint 3 Passed: A box plot has been defined!"
##
## Problem 12
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Recall that the box plot summarizes the distribution in five numbers: the minimum, the first quartile (with 25% of the data below it), the median, the third quartile (with 75% of the data below it), and the maximum. Each of these numbers has at least one corresponding R function:

Number	R function
Minimum	`min(variable)`
First quartile	`quantile(variable, probs = 0.25)`
Median	`median(variable)` or `quantile(variable, probs = 0.5)`
Third quartile	`quantile(variable, probs = 0.75)`
Maximum	`max(variable)`

13. [2 points] Use a combination of `dplyr`'s `summarize` function and the above functions to compute the five number summary of `CS_rate_100`. Assign the summary to the name `five_num_summary`, which should contain values for `min`, `Q1`, `median`, `Q3`, and `max`.

```
five_num_summary <- CS_data %>% summarize(min = min(CS_rate_100),
Q1 = quantile(CS_rate_100, 0.25), median = median(CS_rate_100), Q3 = quantile(CS_rate_100, 0.75),
max = max(CS_rate_100))
five_num_summary
```

```
## # A tibble: 1 x 5
##     min    Q1 median    Q3   max
##   <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1   0.4   5.1   15.6  23.3  45.9
```

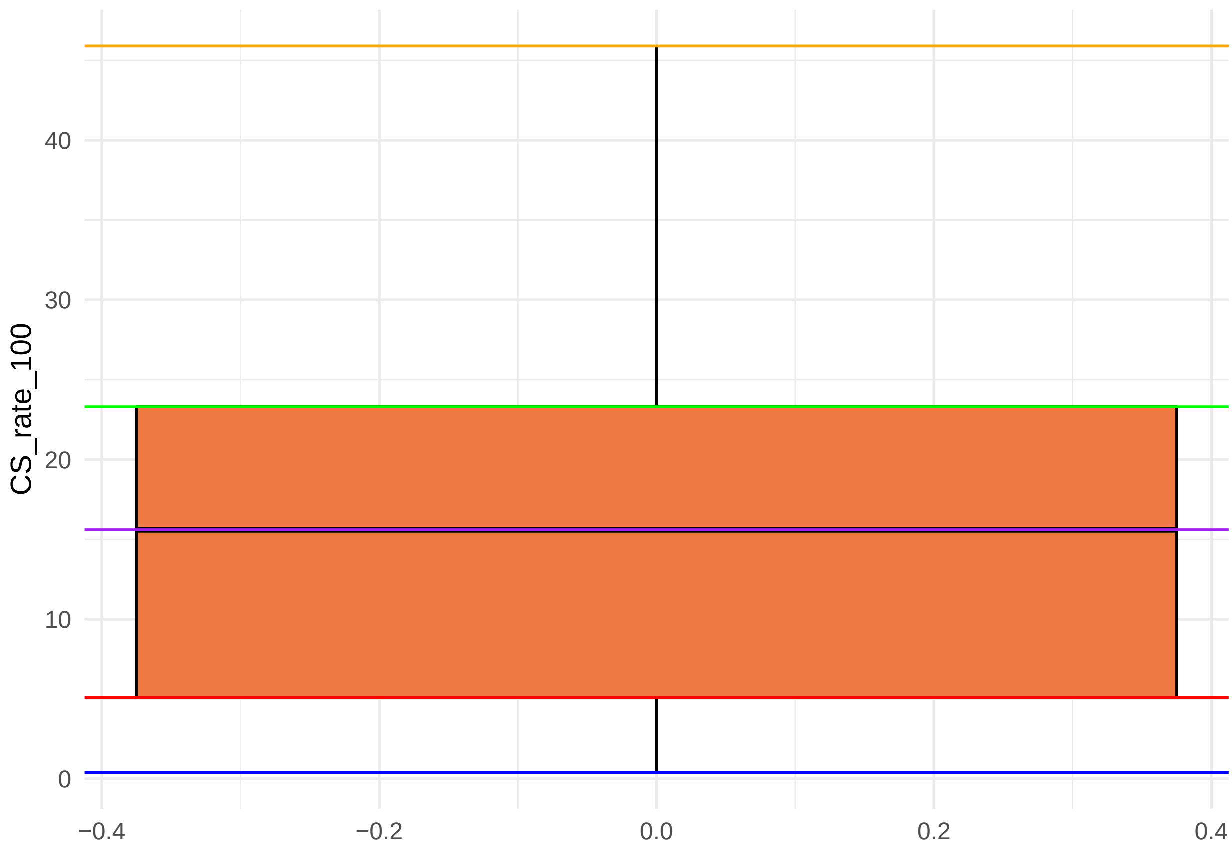
```
check_problem13()
```

```
## [1] "Checkpoint 1 Passed: Correct min value has been computed!"
## [1] "Checkpoint 2 Passed: Correct value of the first quartile has been computed!"
## [1] "Checkpoint 3 Passed: Correct median value has been computed!"
## [1] "Checkpoint 4 Passed: Correct value of the third quartile has been computed!"
## [1] "Checkpoint 5 Passed: Correct max value has been computed"
##
## Problem 13
## Checkpoints Passed: 5
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Double check that `geom_boxplot()` is making the box plot correctly. You can do this by adding horizontal lines to the plot at each number in your five number summary using `geom_hline()`. Because horizontal lines intercept the y-axis, `geom_hline()` requires the `yintercept` argument that you can set to each number in your summary.

14. [2 points] The code below includes one horizontal line at the minimum shown in blue. Add the rest of the lines:

```
p14 <- ggplot(data = CS_data, aes(y = CS_rate_100)) +
          geom_boxplot(col = "black", fill = "sienna2") +
          theme_minimal() +
          geom_hline(aes(yintercept = 0.4), col = "blue") +
          geom_hline(aes(yintercept = 5.1), col = "red") +
          geom_hline(aes(yintercept = 15.6), col = "purple") +
          geom_hline(aes(yintercept = 23.3), col = "green") +
          geom_hline(aes(yintercept = 45.9), col = "orange")
p14
```




```
check_problem14()
```

```
## [1] "Checkpoint 1 Passed: A geom_hline has been defined!"
## [1] "Checkpoint 2 Passed: A geom_hline has been defined!"
## [1] "Checkpoint 3 Passed: A geom_hline has been defined!"
## [1] "Checkpoint 4 Passed: A geom_hline has been defined!"
## [1] "Checkpoint 5 Passed: A geom_hline has been defined!"
##
## Problem 14
## Checkpoints Passed: 5
```

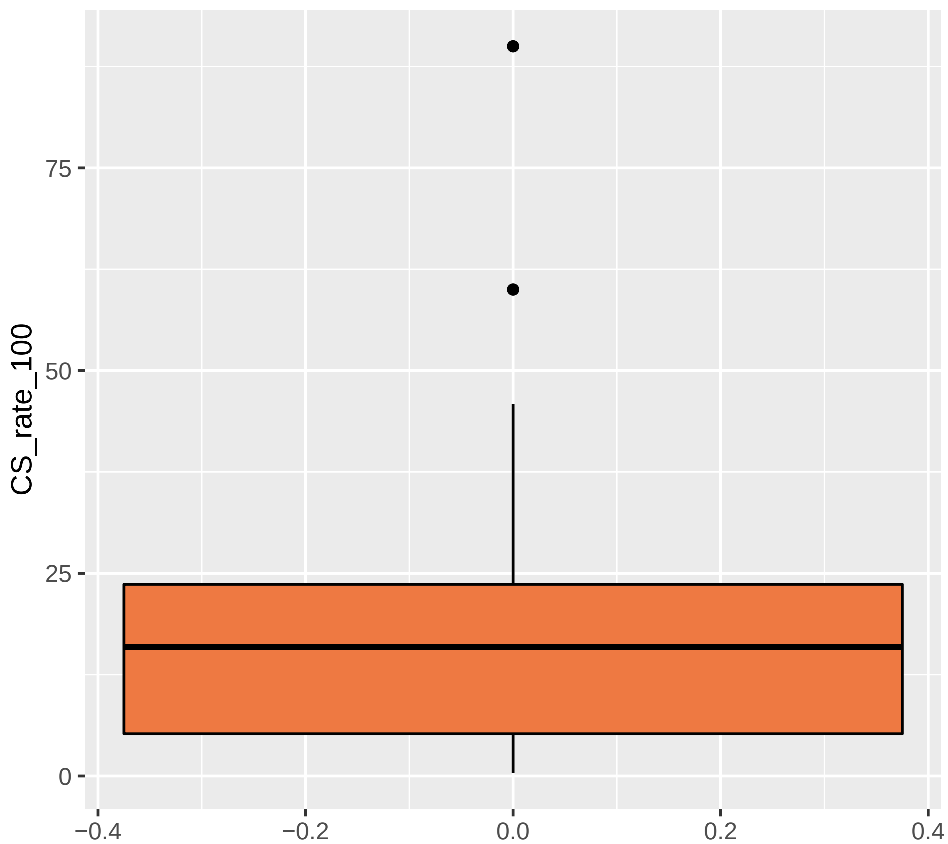
```
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

15. [4 marks] Compile the following code which adds two points to the CS__data, makes a new dataset called `CS_data_plus_2`, and redraws the box plot. How did the box plot change? Perform a calculation to justify why it changed. What are the newly-added features on the plot called?

```
out_data <- tibble::tribble(
  ~Country_Name, ~CS_rate_100,
  "Point 1", 90,
  "Point 2", 60
)

CS_data_plus_2 <- dplyr::full_join(CS_data, out_data)
```

```
## Joining, by = c("Country_Name", "CS_rate_100")
```

```
ggplot(data = CS_data_plus_2, aes(y = CS_rate_100)) +
  geom_boxplot(col = "black", fill = "sienna2")
```




```
new_IQR <- CS_data_plus_2 %>% summarize(min = min(CS_rate_100),
Q1 = quantile(CS_rate_100, 0.25), median = median(CS_rate_100), Q3 = quantile(CS_rate_100, 0.75),
max = max(CS_rate_100))
new_IQR
```

```
## # A tibble: 1 x 5
##     min    Q1 median    Q3   max
##   <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1   0.4  5.20   15.9  23.6    90
```

The Q1, median, Q3, and max of the dataset all increased. The newly-added features to the box plot are called outliers. These outliers shifted all the quartile values and max higher. The box plot got flatter/more “squished,” and the two new points were also plotted at the top of the box plot.

Check your score

Click on the middle icon on the top right of this code chunk (with the downwards gray arrows 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                 2    autograded
## Problem 2           PASSED                 1    autograded
## Problem 3           PASSED                 1    autograded
## Problem 4  NOT YET GRADED                 3 free-response
## Problem 5           PASSED                 1    autograded
## Problem 6           PASSED                 2    autograded
## Problem 7           PASSED                 1    autograded
## Problem 8           PASSED                 1    autograded
## Problem 9  NOT YET GRADED                 2 free-response
## Problem 10 NOT YET GRADED                 1 free-response
## Problem 11          PASSED                 1    autograded
## Problem 12          PASSED                 2    autograded
## Problem 13          PASSED                 2    autograded
## Problem 14          PASSED                 2    autograded
## Problem 15 NOT YET GRADED                 4 free-response
```