Using Logical Vectors with `summarise()`

Details

The Summaries section of the Logical Vectors chapter in R for Data Science has three sections:

1. Logical Summaries
2. Numeric Summaries
3. Logical Subsetting

For each section, your group should produce a summary that includes:

- A description of any new functions that are introduced along with a small example that demonstrate their use on a logical vector. *(This can include functions or arguments that are used in the code examples, but aren't the primary focus of the section)*.
- An explanation of the code example used in the section in your own words.
- A summary of when the technique that is presented might be useful.

Consider your group mates (or perhaps your future self) as the audience for your summary. Include the details you thought were missing on your first reading, or any intermediate code you ran to understand the examples.

Logical Summaries

1. There are two functions within logical summaries, any() and all():

   any() will tell us if at least one of the values of the vector is TRUE. It is like the or | expression we just saw in the previous examples.

   all() will tell us if all the values of the vector are TRUE and is similar to the and & expression.

   Using the na.rm = FALSE as the second argument in either function will remove the missing values.

```
starwars |>
  group_by(species) |>
  summarize(
```

All_Mass_Small = all(mass <= 60, na.rm = TRUE),

Any_Age_Young = any(birth_year > 100, na.rm = TRUE))

This example will tell you which of the species all had a small mass of less than 60g and also were born after the year 100.

2. We can use logical summaries to work through a data frame to answer questions about the data like if a flight was delayed by a certain time, mass a species and after a year as seen in the example above.

3. This technique presented can be useful when wanting a crude quick answer to the data without much extra detail (just getting a true or false answer).

Numeric Summaries

1. The summarise() function can include various summary functions like mean(), median(), sum(), etc. for creating summary statistics to analyze the data. Let us consider an example with a logical vector.

library(dplyr)

Data <- data.frame(x = c(5,10,15,20,25,30,35,40,50))

Threshold_value <- 15

Logical_vector <- data$x > Threshold_value

Summary_statistics <- Data |> summarise(mean_greater_than_Threshold_value = mean(Logical_vector) , sum_greater_than_Threshold_value = sum(Logical_vector)

print(Summary_statistics)

2. The data is a data frame with a numeric vector assigned to variable “x”. The threshold value is set to 15. The logical vector is created by comparing if each value in “x” is greater than the threshold value. The summarise() function is used to calculate the mean and sum of the logical vector greater than threshold value.

3. It might be useful when we want to estimate the proportion of true values in a logical vector, we can use mean() in that case. When we want to count the number of occurrences where a condition is true in a logical vector, we can use median().

Logical Subsetting

1. This section formally introduces the n() function, although we've seen it in previous sections. The n() function works in tandem with the group_by() function to return the number of elements in each group. Much in the way that aes() is a helper function to ggplot() commands, n() is a helper function for mutate(), filter(), and summarize(). As an example:

```
library(tidyverse)

starwars |>

  group_by(species) |>

  summarize(total = n())
```

Would return the number of characters of each species in the starwars dataset.

2. We can use logical vectors to select subsets of data within our summaries. For example, suppose I was working with income data of a town and I wanted to know the average incomes for the bottom 90% and the top 10% of income earners and how they change over time. If it was only one of these, I could simply use the filter function to remove the other group, but I can't do both at the same time with filter(). In place of filter(), I can use logical vectors:

```
incomedata |>

  group_by(year) |>

  summarize(

        avg_b90 = mean(income[income < 101500], na.rm = TRUE),

        avg_t10 = mean(income[income >= 101500], na.rm = TRUE)

  )
```

Avg_b90 is the mean of all incomes under 101,500 (which is the hypothetical break-point between the bottom 90% and the top 10%), while avg_t10 is the mean of all incomes above 101,500.

3. This technique can be used when wanting to answer further questions about the data, using a logical vector to filter a single variable.