Lab1_122

2025-04-02

Your Turn:

1a.

```
x <- c(10,12,5,14,35)
print(x)
```

```
## [1] 10 12  5 14 35
```

```
mean(x)
```

```
## [1] 15.2
```

1b.

In your R Markdown file, write a for loop to do the following: • Use the sample function to sample from x five times, WITH replacement • Find the mean of this new sample, and store it into the i th spot of a vector. – Hint: you can do this by either storing the sample from the previous step as a variable and taking the mean of that, or wrap the mean function around your code from the previous step and doing it all in one line. • Do this 10 times, and print the result

```
set.seed(422025)
reps <- 10
output <- NA
for(i in 1:reps)
{
  sample_set <- sample(x, size = 5, replace = TRUE)
  output[i] <- mean(sample_set)
}
print(output)
```

```
##  [1] 10.6 12.4 14.8  9.4 19.8 13.8 10.2 15.6 16.2 14.2
```

1c.

Write a function to do the following: • sample 3 elements from a vector, WITHOUT replacement (you can assume that the input vector has at least 3 elements; you do not need to make your function check for it) • Calculate the standard deviation of the sample of 3 elements, using the sd function Put this in your R Markdown file, and run it on the vector x that we have been using.

```r
samp_sd <- function(x)
{
  new_samp <- sample(x, size = 5, replace = FALSE)
  return(sd(new_samp))
}
samp_sd(x)
```

```
## [1] 11.56287
```

2.

The data for this question come from an experimental study in which researchers studied the impact of participating in wellness programs with therapy dogs on emotional wellbeing (Binfet et al., 2022). A sample of 284 college students were randomly assigned into different treatment groups that dictated exactly what their contact with the therapy dog would be, and their emotional wellbeing was subsequently measured. We will investigate this study in more detail in future labs, but for now, we will practice reading in the file and obtaining some descriptive statistics. The TAs will run through this question together with you in the Discussion Sections.

1) First, download the dog_data_cleaned.csv file from Canvas, and put it into the same directory as your Rmd file for this lab.
2) Then go to the "Import" tool, "From text (readr)." Click on "Browse" and find the dog_data_cleaned.csv file.
3) If the preview looks good, click on "Import" so that your active R session has the dataset loaded and you can interact with it there as needed.
4) We also want to get it loaded into our R Markdown file. As long as you have the data file in the same directory as your Rmd file, then putting just the following in a code chunk should do it:

```r
library(readr)
dog_data_cleaned <- read_csv("dog_data_cleaned.csv")
```

```
## Rows: 568 Columns: 16
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): GroupAssignment, Stage
## dbl (14): RID, Age_Yrs, Year_of_Study, Live_Pets, Consumer_BARK, PANAS_PA, S...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2a.

The dataset has two rows for each individual: a pre and post test value. For this question, we want to focus on the pre test values, so let's filter the dataset to just those rows using tidyverse tools. If you have never used the tidyverse, you'll first want to run in your R Console:

install.packages("tidyverse")

Reminder: do this in your CONSOLE ONLY; do NOT put this in your R Markdown file! It will take a while to run, and you don't want to have it run this every single time you knit your Rmd file! Then, write dplyr code in your R Markdown file to create a new filtered dataset that only contains rows that are the pre test values.

```
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
dog_pre <- dog_data_cleaned %>% filter(Stage == "pre")
dog_pre
```

```
## # A tibble: 284 x 16
##      RID GroupAssignment Age_Yrs Year_of_Study Live_Pets Consumer_BARK Stage
##    <dbl> <chr>             <dbl>         <dbl>     <dbl>         <dbl> <chr>
##  1     1 Control              21             3         2             1 pre
##  2     2 Direct               19             1         2             1 pre
##  3     3 Indirect             18             1         2             1 pre
##  4     4 Control              18             1         2             1 pre
##  5     5 Direct               19             1         2             1 pre
##  6     6 Indirect             20             2         2             1 pre
##  7     7 Control              26             2         1             1 pre
##  8     8 Direct               17             1         2             1 pre
##  9     9 Indirect             21             3         2             1 pre
## 10    10 Control              22             4         2             1 pre
## # i 274 more rows
## # i 9 more variables: PANAS_PA <dbl>, SHS <dbl>, SCS <dbl>, Engagement <dbl>,
## #   FS <dbl>, Stress <dbl>, Homesick <dbl>, Lonely <dbl>, PANAS_NA <dbl>
```

2b.

Find/calculate the following descriptive statistics for the Lonely variable: • n - the number of observations for this variable • x - the sample mean of x where x is Lonely • sx - the sample standard deviation of x Put each of these values into a kable table (from the knitr package) with appropriate column headings, and also put them into a sentence in the text of your R Markdown file using R code in flow.

```
library(knitr)
stats_tidy <- dog_pre %>% summarize(n = n(),
                                    x_bar = round(mean(Stress), 2),
                                    sx = round(sd(Stress), 2))
kable(stats_tidy)
```

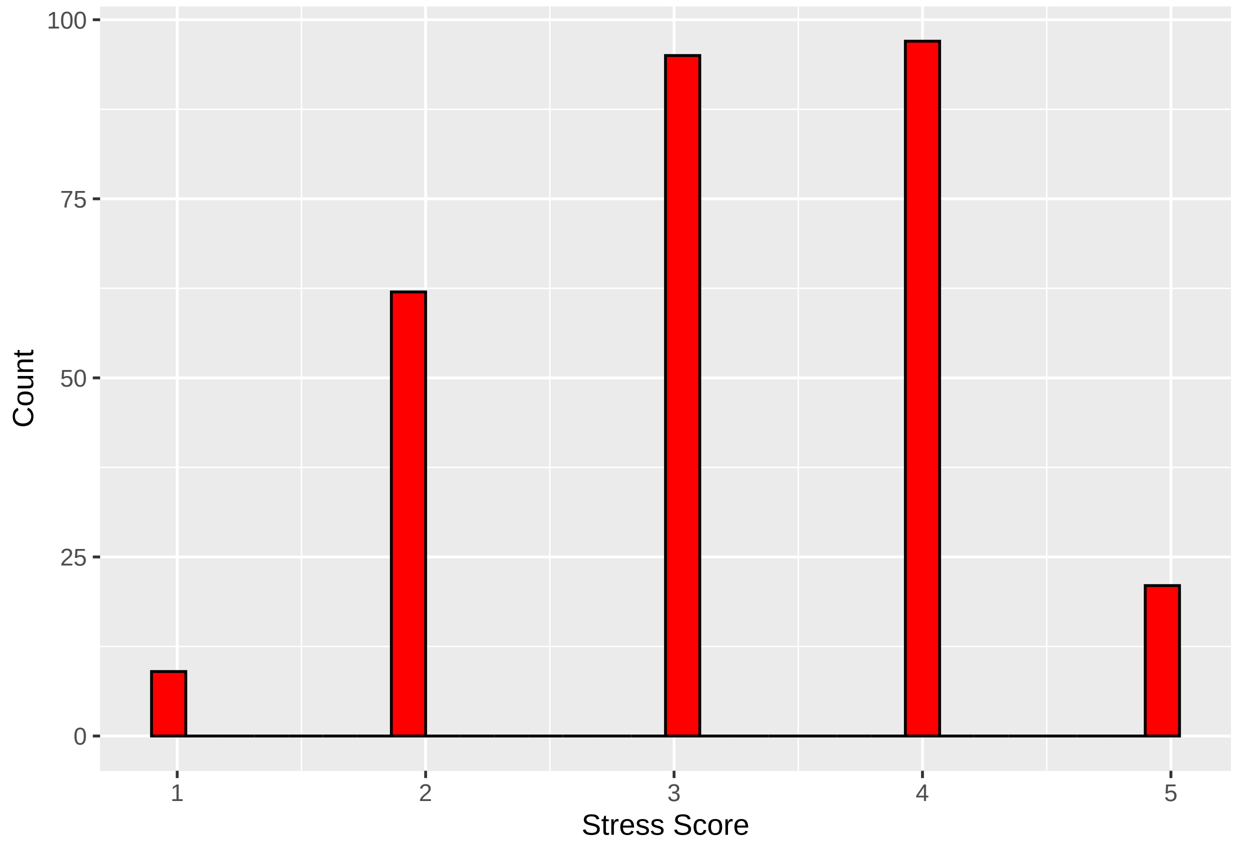
n	x_bar	sx
284	3.21	0.97

2c.

Make a histogram of the Lonely variable using both base R plotting and ggplot2. In both cases, be sure to change/make axis labels and title as needed.

```r
ggplot(dog_pre, aes(x = Stress)) +
  geom_histogram(fill = "red", color = "black") +
  xlab("Stress Score") +
  ylab("Count")
```

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



```r
  ggtitle("Histogram of Stress Scores")
```

```
## $title
## [1] "Histogram of Stress Scores"
##
## attr(,"class")
## [1] "labels"
```

3

Now load the dog_data_post.csv dataset, using steps similar to the above. This dataset contains the same variables as in the previous dataset, but with their post-treatment minus pre-treatment differences. Then, do the following:

```r
library("dplyr")
dog_data_post <- read_csv("dog_data_post.csv")
```

```
## Rows: 284 Columns: 18
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): GroupAssignment, Stage, Group
## dbl (15): RID, Age_Yrs, Year_of_Study, Live_Pets, Consumer_BARK, PANAS_PA, S...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

3a.

Create a new filtered dataset that has only those that are in the control group (that is, those with a value of Control for the variable GroupAssignment).

```r
library("dplyr")
control_group <- dog_data_post %>% filter(GroupAssignment == "Control")
head(control_group)
```

```
## # A tibble: 6 x 18
##     RID GroupAssignment Age_Yrs Year_of_Study Live_Pets Consumer_BARK Stage
##   <dbl> <chr>             <dbl>         <dbl>     <dbl>         <dbl> <chr>
## 1     1 Control              21             3         2             1 post
## 2     4 Control              18             1         2             1 post
## 3     7 Control              26             2         1             1 post
## 4    10 Control              22             4         2             1 post
## 5    13 Control              19             2         2             1 post
## 6    16 Control              17             1         2             2 post
## # i 11 more variables: PANAS_PA <dbl>, SHS <dbl>, SCS <dbl>, Engagement <dbl>,
## #   FS <dbl>, Stress <dbl>, Homesick <dbl>, Lonely <dbl>, PANAS_NA <dbl>,
## #   Diff <dbl>, Group <chr>
```

3b

Find the same descriptive statistics as in part (b) from the previous question but now for the filtered dataset that you just created, and make a kable table of them. Also type a sentence stating the values with R code in flow to show the values.

```r
library(knitr)
n_lonely <- nrow(control_group)
mean_lonely <- mean(control_group$Lonely, na.rm = TRUE)
sd_lonely <- sd(control_group$Lonely, na.rm = TRUE)
```

```r
statstablecontrol <- data.frame(
  Statistic = c("n", "Mean", "Standard Deviation"),
  Value = c(as.integer(n_lonely), round(mean_lonely, 2), round(sd_lonely, 2))
)

kable(statstablecontrol, col.names = c("Statistic", "Value"), caption = "Lonely Variable in Control Grou
```

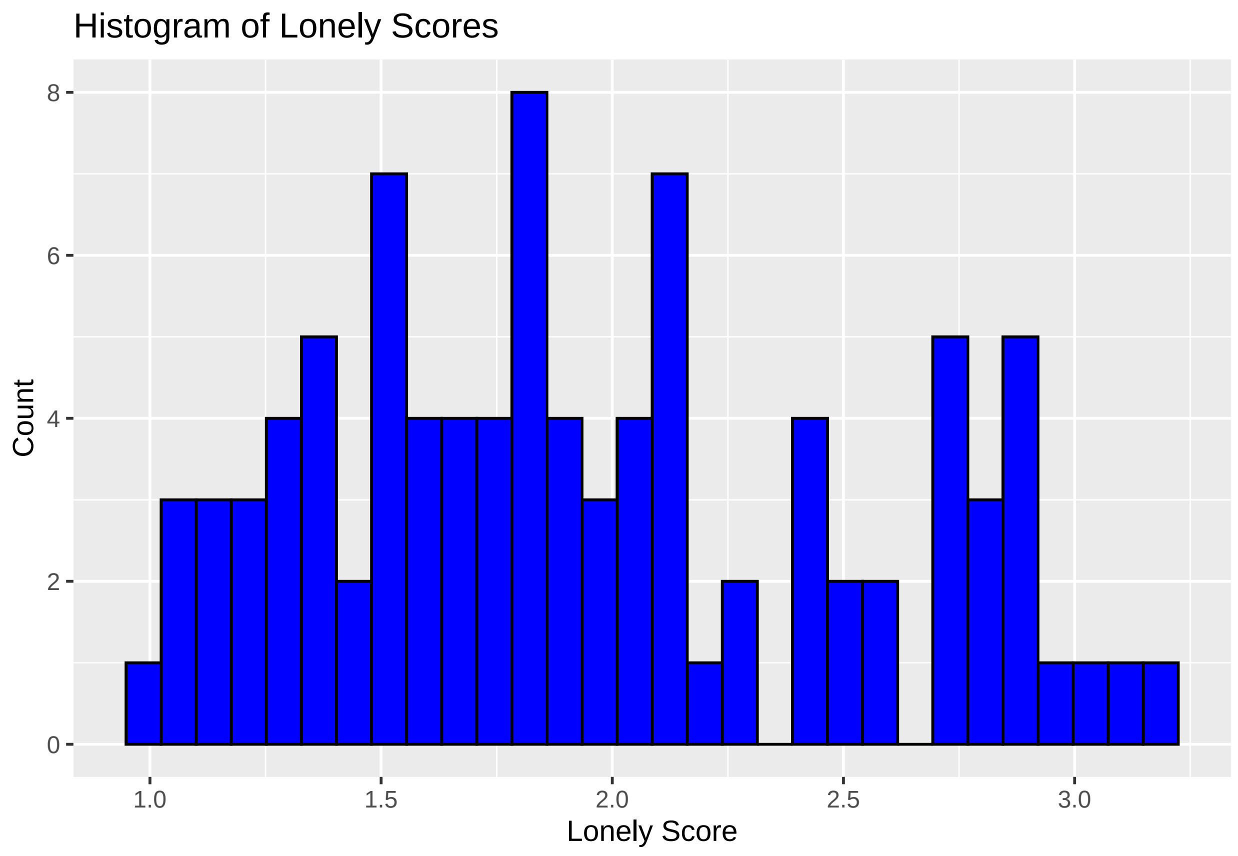
Table 2: Lonely Variable in Control Group

Statistic	Value
n	94.00
Mean	1.96
Standard Deviation	0.57

3c

Make a histogram of the Lonely variable among the control group using both base R plotting and ggplot2. Again be sure to change/make axis labels and title as needed.

```r
ggplot(control_group, aes(x = Lonely)) +
  geom_histogram(bins = 30, fill = "blue", color = "black") +
  xlab("Lonely Score") +
  ylab("Count") +
  ggtitle("Histogram of Lonely Scores")
```



4

$$f(x) = \sum_{i=1}^{n} x_i$$

4a

Write the following: A simple linear regression model is y = 0 + 1x

$$y = \beta_0 + \beta_1 x$$

4b

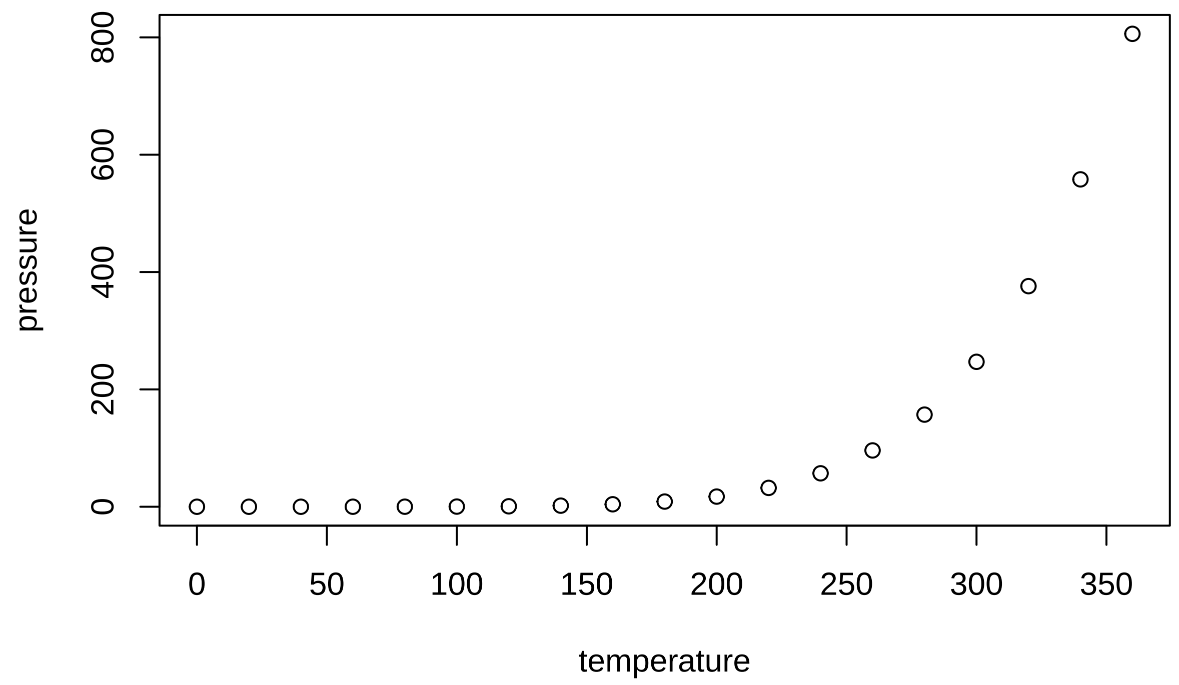
Write the following: The formula for the sample standard deviation is:

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}}$$

4c

Write the following as aligned equations: (see the first link for help)

$$\begin{aligned} H_0: &\quad \mu = \mu_0 \\ H_A: &\quad \mu \neq \mu_0 \end{aligned}$$



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.