DSCI 552 Lab 1

Populations, Sampling, Bootstrapping, and Sampling Distribution

Contents

Lab Mechanics

rubric={mechanics:5}

- Paste the URL to your GitHub repo here: **https://github.ubc.ca/MDS-2022-23/DSCI_552_lab1_kewang5**
- Once you finish the assignment, you must **knit** this `R` markdown to create a `.pdf` file and push everything to your GitHub repo using `git push`. You are responsible for ensuring all the figures, texts, and equations in the `.pdf` file are appropriately rendered.
- You must submit this `.Rmd` **and** the rendered `.pdf` files to Gradescope.

  Heads-up: You need to have a minimum of 3 commits.

Code Quality

rubric={quality:3}

The code that you write for this assignment will be given one overall grade for code quality. Check our **code quality rubric** as a guide to what we are looking for. Also, for this course (and other MDS courses that use `R`), we are trying to follow the `tidyverse` code style. There is a guide you can refer too: http://style.tidyverse.org/

Each code question will also be assessed for code accuracy (i.e., does it do what it is supposed to do?).

Writing

rubric={writing:3}

To get the marks for this writing component, you should:

- Use proper English, spelling, and grammar throughout your submission (the non-coding parts).
- Be succinct. **This means being specific about what you want to communicate, without being superfluous.**

Check our **writing rubric** as a guide to what we are looking for.

Setup

If you fail to load any packages, you can install them and try loading the library again.

```
library(cowplot)
library(infer)
library(knitr)
library(tidyverse)
library(digest)
library(datateachr)
library(testthat)
```

Note you need to install the package `datateachr` via the following:

1. Uncomment the two lines of code below by deleting the `#` at the start of each line.
2. Run the code cell, which will perform the installation.
3. Comment the two lines of code again by adding the `#` back to the start of each line.

```
# options(timeout=9999999)
# devtools::install_github("UBC-MDS/datateachr")
```

Exercise 1: Conceptual Warmup

rubric={autograde:3}

Read the mixed-up table below and assign each object in the code cell below the integer associated with its correct definition.

> **Note:** Some of these terms may have different meanings in other fields different from Statistics, but these are the definitions that we will be using all over this course and subsequent ones.

Terms	Definitions
Point estimate	**1.** The entire set of entities objects of interest.
Population	**2.** A numerical summary value about the population.
Population parameter	**3.** A collected subset of observations from a population.
Sample	**4.** A summary statistic calculated from a random sample that estimates an unknown population parameter of interest.
Observation	**5.** A distribution of point estimates, where each point estimate was calculated from a different random sample coming from the same population.
Sampling distribution	**6.** A quantity or quality (or a set of these) from a single member of a population.

Assign your answers to the objects given below (`point_estimate`, `population`, `population_parameter`, `sample`, `observation`, and `sampling_distribution`). Your answers should each be a numeric vector of length one (e.g., `term <- 9`).

> **Heads-up:** There are hidden tests which are not shown here and will be applied after you submit.

```
point_estimate <- 4
population <- 1
population_parameter <- 2
sample <- 3
observation <- 6
sampling_distribution <- 5

# YOUR CODE HERE
```

```
. = ottr::check("tests/Q1.R")
```

```
## Test Q1 - 1 passed
##
##
## Test Q1 - 2 passed
##
##
## Test Q1 - 3 passed
##
##
## Test Q1 - 4 passed
##
##
## Test Q1 - 5 passed
##
##
## Test Q1 - 6 passed
```

The Dataset

In this lab, we will explore the population and sampling distributions of one of three different populations of trees planted in Vancouver. To do this, we will use the `vancouver_trees` data set, which includes information about the entire population of public trees planted along boulevards in Vancouver, such as their approximate height, diameter, family and species name, and other information describing where and when they were planted.

This data set is originally from the **City of Vancouver's Open Data Portal**, but we have included it in an R package called **`datateachr`**. The `datateachr` package contains several open source data sets compiled from various sources to make them easily accessible. Let us take a look at the first few rows of the `vancouver_trees` data set.

```
str(vancouver_trees)
```

```
## tibble [146,611 x 20] (S3: tbl_df/tbl/data.frame)
##  $ tree_id           : num [1:146611] 149556 149563 149579 149590 149604 ...
##  $ civic_number      : num [1:146611] 494 450 4994 858 5032 ...
##  $ std_street        : chr [1:146611] "W 58TH AV" "W 58TH AV" "WINDSOR ST" "E 39TH AV" ...
##  $ genus_name        : chr [1:146611] "ULMUS" "ZELKOVA" "STYRAX" "FRAXINUS" ...
##  $ species_name      : chr [1:146611] "AMERICANA" "SERRATA" "JAPONICA" "AMERICANA" ...
##  $ cultivar_name     : chr [1:146611] "BRANDON" NA NA "AUTUMN APPLAUSE" ...
##  $ common_name       : chr [1:146611] "BRANDON ELM" "JAPANESE ZELKOVA" "JAPANESE SNOWBELL" "AUTUMN AI
##  $ assigned          : chr [1:146611] "N" "N" "N" "Y" ...
##  $ root_barrier      : chr [1:146611] "N" "N" "N" "N" ...
##  $ plant_area        : chr [1:146611] "N" "N" "4" "4" ...
##  $ on_street_block   : num [1:146611] 400 400 4900 800 5000 500 4900 4900 4900 700 ...
##  $ on_street         : chr [1:146611] "W 58TH AV" "W 58TH AV" "WINDSOR ST" "E 39TH AV" ...
##  $ neighbourhood_name: chr [1:146611] "MARPOLE" "MARPOLE" "KENSINGTON-CEDAR COTTAGE" "KENSINGTON-CEDA
##  $ street_side_name  : chr [1:146611] "EVEN" "EVEN" "EVEN" "EVEN" ...
##  $ height_range_id   : num [1:146611] 2 4 3 4 2 2 3 3 2 2 ...
##  $ diameter          : num [1:146611] 10 10 4 18 9 5 15 14 16 7.5 ...
##  $ curb              : chr [1:146611] "N" "N" "Y" "Y" ...
##  $ date_planted      : Date[1:146611], format: "1999-01-13" "1996-05-31" ...
##  $ longitude         : num [1:146611] -123 -123 -123 -123 -123 ...
##  $ latitude          : num [1:146611] 49.2 49.2 49.2 49.2 49.2 ...
```

Exercise 2: The Sampling Distribution of the Mean

To begin, we will look at the population of Acer (labelled `ACER` in the data set) trees planted along streets in Vancouver. The Acer genus (or family) of trees are commonly referred to as maple trees, and there are 31 different species currently planted throughout the city. Maple trees are popular along streets in Vancouver, making up around 25% of all planted trees in the city. They are well known for their bright shades of red, orange, and yellow during the fall and for the appearance of a maple leaf on the Canadian flag.

Figure 1: Acer Tree.

For this exercise, we want to estimate the mean diameter of Acer trees, $\hat{\mu}_{\text{diameter}}$, by drawing a sample of $n = 100$ Acer trees from our data set. The diameter in the data set is in **inches**, but we use the metric system in Canada, so make sure to convert the diameter to **meters**.

Q2.1.

rubric={autograde:2}

Filter for `ACER` trees and their diameters. The output **data frame** `answer2_1` should have only 1 column `diameter`. Make sure to convert to the metric system.

```
vancouver_trees |> head()
```

```
## # A tibble: 6 x 20
##   tree_id civic_number std_str~1 genus~2 speci~3 culti~4 commo~5 assig~6 root_~7
##     <dbl>        <dbl> <chr>     <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
## 1  149556          494 W 58TH AV ULMUS   AMERIC~ BRANDON BRANDO~ N       N
## 2  149563          450 W 58TH AV ZELKOVA SERRATA <NA>    JAPANE~ N       N
## 3  149579         4994 WINDSOR ~ STYRAX  JAPONI~ <NA>    JAPANE~ N       N
## 4  149590          858 E 39TH AV FRAXIN~ AMERIC~ AUTUMN~ AUTUMN~ Y       N
## 5  149604         5032 WINDSOR ~ ACER    CAMPES~ <NA>    HEDGE ~ N       N
## 6  149616          585 W 61ST AV PYRUS   CALLER~ CHANTI~ CHANTI~ N       N
## # ... with 11 more variables: plant_area <chr>, on_street_block <dbl>,
## #   on_street <chr>, neighbourhood_name <chr>, street_side_name <chr>,
## #   height_range_id <dbl>, diameter <dbl>, curb <chr>, date_planted <date>,
## #   longitude <dbl>, latitude <dbl>, and abbreviated variable names
## #   1: std_street, 2: genus_name, 3: species_name, 4: cultivar_name,
## #   5: common_name, 6: assigned, 7: root_barrier
```

```
# 1 inch = 0.0254 meters
acer_diameter <- vancouver_trees |>
  filter(genus_name == "ACER") |>
```

```
  select(diameter) |>
  mutate(diameter = diameter * 0.0254)
acer_diameter
```

```
## # A tibble: 36,062 x 1
##    diameter
##       <dbl>
##  1    0.229
##  2    0.381
##  3    0.356
##  4    0.406
##  5    0.457
##  6    0.260
##  7    0.495
##  8    0.356
##  9    0.248
## 10    0.343
## # ... with 36,052 more rows
```

```
answer2_1 <- acer_diameter
mean(answer2_1$diameter)
```

```
## [1] 0.2694092
```

```
. = ottr::check("tests/Q2.1.R")
```

```
##
## All tests passed!
```

Q2.2.

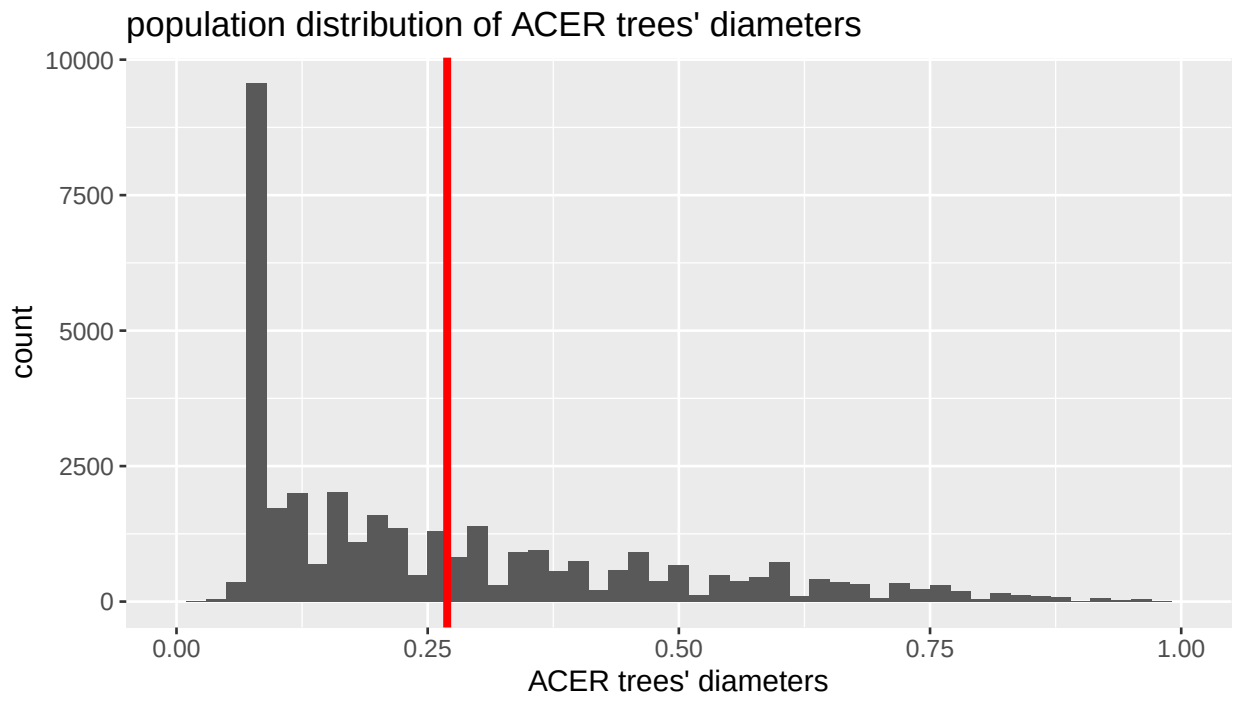
rubric={autograde:2}

Visualize the population distribution of `ACER` trees' diameters stored in `answer2_1` as a histogram. Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to an object called `acer_pop_plot`.

```
acer_pop_plot <- acer_diameter |>
  ggplot(aes(x = diameter)) +
  geom_histogram(binwidth = 0.02) +
  geom_vline(xintercept = mean(acer_diameter$diameter), color = "red", size = 1.5) +
  xlim(0, 1) +
  xlab("ACER trees' diameters") +
  ggtitle("population distribution of ACER trees' diameters") +
  theme(text = element_text(size = 12))

acer_pop_plot
```

```
## Warning: Removed 148 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```




```
. = ottr::check("tests/Q2.2.R")
```

```
##
## All tests passed!
```

Q2.3.

rubric={autograde:3}

Draw a sample of size $n = 100$ from the `ACER` trees population stored in `answer2_1` using appropriate function from the `infer` library. The output data frame `answer2_3` should have two columns:

- `replicate` (which represents the sample number), and
- `diameter` (in meters).

Set seed to `552` so your simulation can be reproducible.

```
set.seed(552)
sample_1 <- rep_sample_n(acer_diameter, size = 100, reps = 1)
answer2_3 <- sample_1
answer2_3
```

```
## # A tibble: 100 x 2
## # Groups:   replicate [1]
##    replicate diameter
##        <int>    <dbl>
##  1         1   0.318
##  2         1   0.305
##  3         1   0.0762
##  4         1   0.0762
##  5         1   0.0762
##  6         1   0.254
##  7         1   0.178
##  8         1   0.0889
```

```
##  9         1   0.190
## 10         1   0.152
## # ... with 90 more rows
```

```
. = ottr::check("tests/Q2.3.R")
```

```
##
## All tests passed!
```

Q2.4.

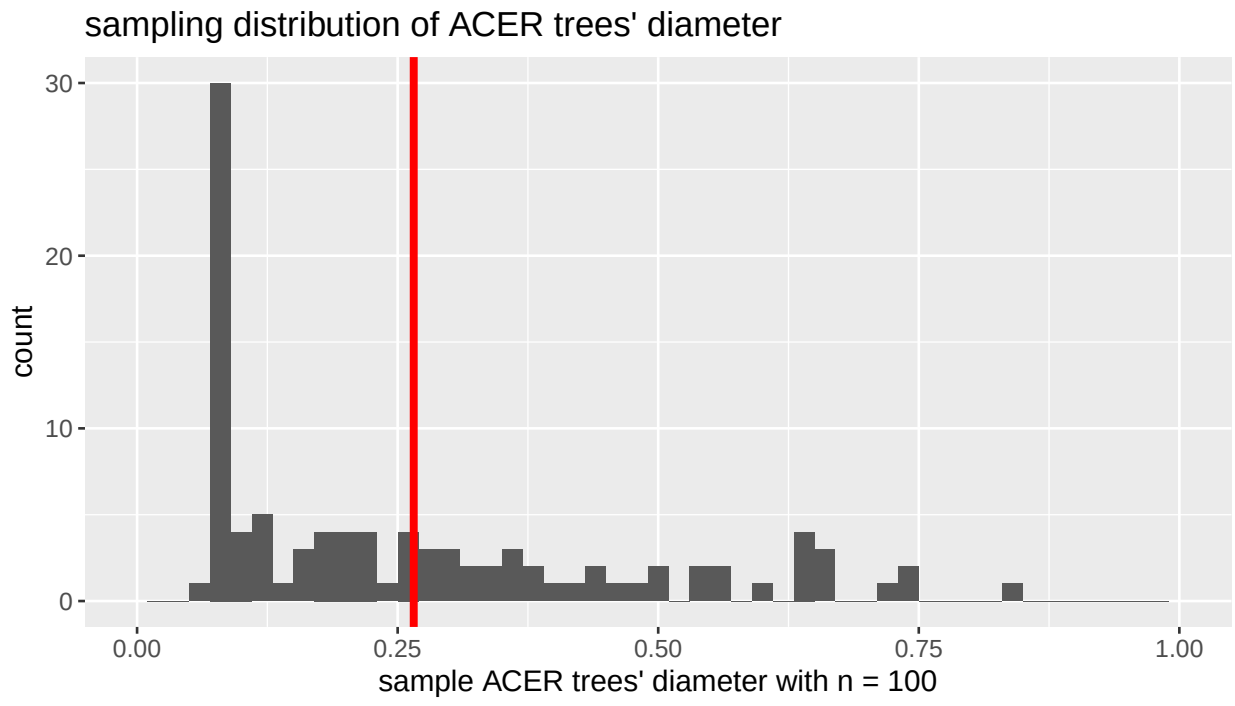
rubric={autograde:2}

Visualize as a histogram the distribution of `ACER` trees' diameters stored in the sample `answer2_3`.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to an object called `single_acer_sample_plot`.

```
single_acer_sample_plot <- sample_1 |>
  ggplot(aes(x = diameter)) +
  geom_histogram(binwidth = 0.02) +
  geom_vline(xintercept = mean(sample_1$diameter), color = "red", size = 1.5) +
  xlim(0, 1) +
  xlab("sample ACER trees' diameter with n = 100") +
  ggtitle("sampling distribution of ACER trees' diameter") +
  theme(text = element_text(size = 12))

single_acer_sample_plot
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```




```
. = ottr::check("tests/Q2.4.R")
```

```
##
```

```
## All tests passed!
```

Q2.5.

rubric={autograde:3}

Draw 10,000 samples of size $n = 100$ from the `ACER` trees population stored in `answer2_1`. For each sample, compute the mean diameter. The output data frame `answer2_5` should have two columns:

- `replicate` (which represents the sample number), and
- `mean` (which represents the mean diameter in meters of each sample). Set seed to `552` so your simulation can be reproducible.

```
set.seed(552)
sample_10000_mean <- acer_diameter |>
  rep_sample_n(size = 100, reps = 10000) |>
  group_by(replicate) |>
  summarise(mean = mean(diameter))
answer2_5 <- sample_10000_mean
answer2_5
```

```
## # A tibble: 10,000 x 2
##    replicate  mean
##        <int> <dbl>
##  1         1 0.265
##  2         2 0.236
##  3         3 0.238
##  4         4 0.281
##  5         5 0.237
##  6         6 0.267
##  7         7 0.316
##  8         8 0.264
##  9         9 0.279
## 10        10 0.268
## # ... with 9,990 more rows
```

```
. = ottr::check("tests/Q2.5.R")
```

```
##
## All tests passed!
```

Q2.6.

rubric={autograde:2}

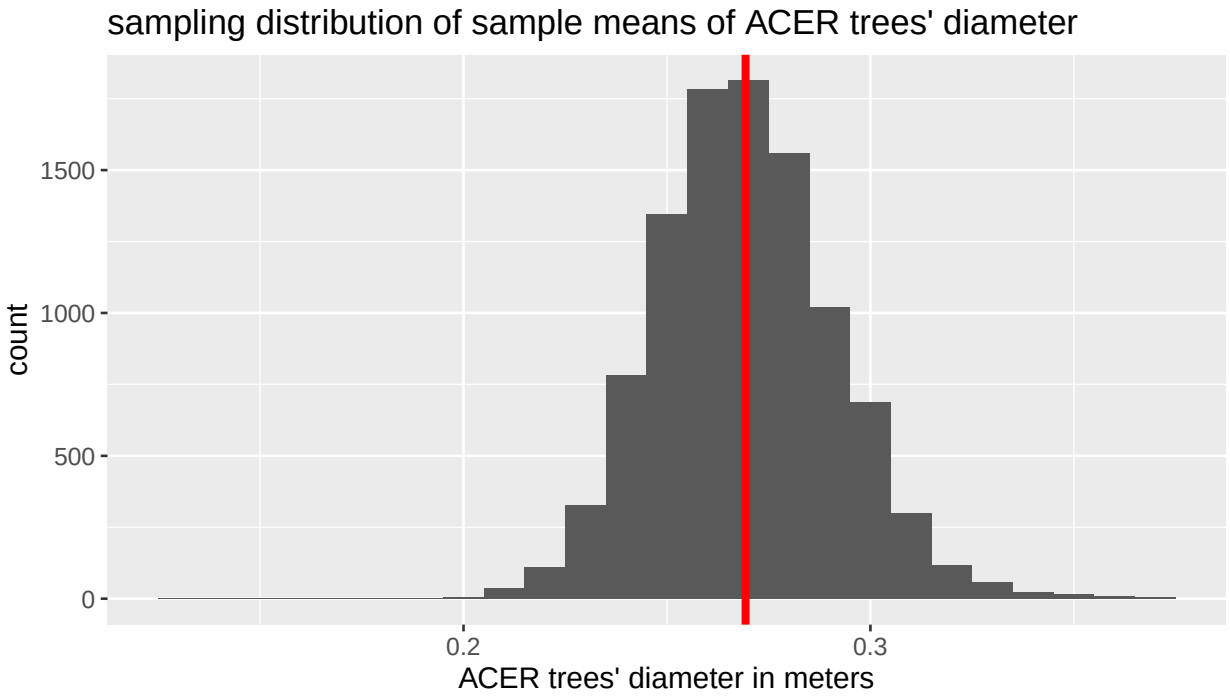
Visualize as a histogram the sampling distribution of sample means (stored in `answer2_5`) of `ACER` tree's diameters from 10000 samples of size $n = 100$.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to an object called `sampling_dist_mean_100_plot`.

```
sampling_dist_mean_100_plot <- sample_10000_mean |>
  ggplot(aes(x = mean)) +
  geom_histogram(binwidth = 0.01) +
  xlim(0.125, 0.375) +
  geom_vline(xintercept = mean(sample_10000_mean$mean), color = "red", size = 1.5) +
  ggtitle("sampling distribution of sample means of ACER trees' diameter") +
  xlab("ACER trees' diameter in meters") +
```

```
  theme(text = element_text(size = 12))
sampling_dist_mean_100_plot
```

```
## Warning: Removed 1 rows containing non-finite values (stat_bin).
```




```
. = ottr::check("tests/Q2.6.R")
```

```
##
## All tests passed!
```

Q2.7.

rubric={viz:2}

Combine the three plots above into one graph (you can use `plot_grid()` from package `cowplot`).

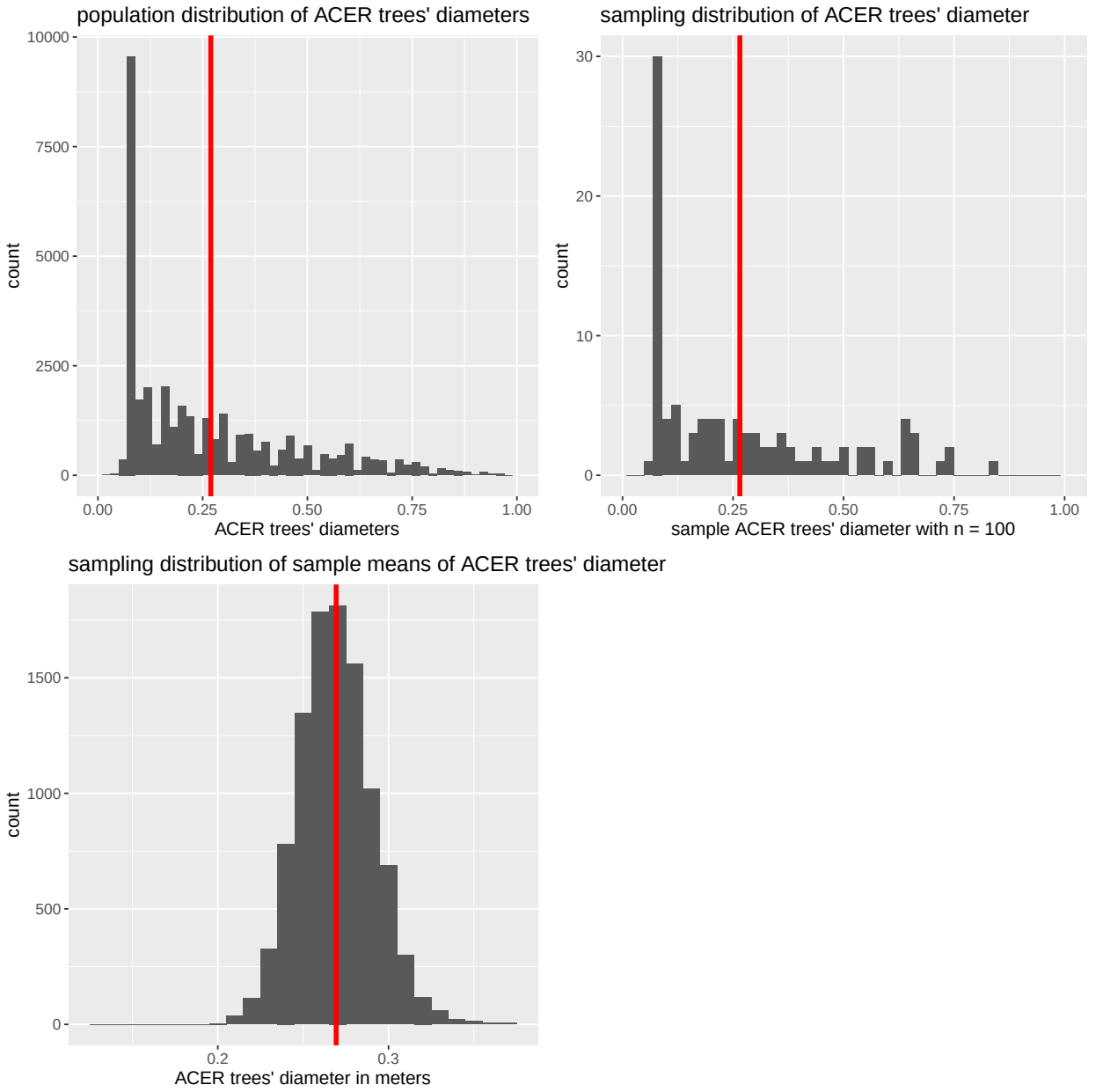
```
all_in_one <-
  plot_grid(acer_pop_plot, single_acer_sample_plot, sampling_dist_mean_100_plot)
```

```
## Warning: Removed 148 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
## Removed 2 rows containing missing values (geom_bar).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_bin).
```

```
all_in_one
```



Q2.8.

rubric={reasoning:10}

Describe how these three previous distributions are similar or different and their relationship to each other. Show how the sample mean $\hat{\mu}_{\text{diameter}}$ in `single_acer_sample_plot` behaves in terms of distribution, spread, and center, when compared to the population mean $\mu_{diameter}$ from `acer_pop_plot`. Do the same for the average of the 10000 means from `sampling_dist_mean_100_plot`.

Write between one and three paragraphs.

The shape (i.e. the distribution and spread) of single sample distribution resambles the population distribution, but the shape of sampling distribution of sample means is different from the previous two. The sampling

distribution of sample means is in bell shape, which resambles the normal distribution. Both sample mean of the single sample and the average of 10000 means are very close to the population mean.

Exercise 3: The Sampling Distribution's Relationship to Sample Size n

We will explore the relationship between sample size n and the shape/spread of the sampling distribution of your estimate. We will use the mean `ACER` tree diameter as our estimate of interest to do this. Using that data, do the following:

Q3.1.

rubric={autograde:5}

Using the population `ACER` diameter data from `answer2_1`, create 3 sampling distributions of the mean by drawing 10,000 samples. Use the sample sizes `10`, `30`, and `100`. For each sampling distribution of the mean, calculate the **mean** and the **standard error** (which is the standard deviation of the sampling distribution of your estimate).

Store the output in a **data frame** called `answer3_1` with the following format:

n	mean	standard_error
10		
30		
100		

You would need to create a function that receives the **population data** and a specific **sample size** as arguments to compute the sample mean and standard deviation **in each one of the 10,000 samples**. Set seed as `552` **as the first line of this function**. Moreover, this function will need to be used **three times** (one per each sample size). Finally, in each of these three times, the function should return the following data frame with 10,000 rows and the following three columns:

- the sample ID number (i.e., `replicate` from `rep_sample_n()`),
- the `mean` computed by sample, and
- the `standard_deviation` computed by sample.

Finally, you can summarize your simulation results in `answer3_1`.

```
head(acer_diameter)
```

```
## # A tibble: 6 x 1
##   diameter
##      <dbl>
## 1    0.229
## 2    0.381
## 3    0.356
## 4    0.406
## 5    0.457
## 6    0.260
```

```
simulate_10k <- function(pop_data, sample_size) {
  set.seed(552)
  pop_data |>
    rep_sample_n(size = sample_size, reps = 10000) |>
    group_by(replicate) |>
    summarise(mean = mean(diameter), standard_deviation = sd(diameter))
}

simulate_10k_n_sample_size <- function(pop_data, sample_size) {
```

```
  pop_data |>
  simulate_10k(sample_size) |>
  select(mean) |>
  summarise(mean_mean = mean(mean), se_mean = sd(mean)) |>
  mutate(n = sample_size, mean = mean_mean, standard_error = se_mean) |>
  select(n, mean, standard_error)
}

result_10 <- acer_diameter |>
  simulate_10k_n_sample_size(10)

result_30 <- acer_diameter |>
  simulate_10k_n_sample_size(30)

result_100 <- acer_diameter |>
  simulate_10k_n_sample_size(100)
```

```
answer3_1 <- bind_rows(result_10, result_30, result_100)
answer3_1
```

```
## # A tibble: 3 x 3
##       n  mean standard_error
##   <dbl> <dbl>          <dbl>
## 1    10 0.270         0.0718
## 2    30 0.269         0.0406
## 3   100 0.269         0.0219
```

```
. = ottr::check("tests/Q3.1.R")
```

```
##
## All tests passed!
```

Q3.2.

rubric={viz:5}

Plot the 3 sampling distributions of the mean one after another (vertically). Store your plots in the variable `sampling_dist_by_n`

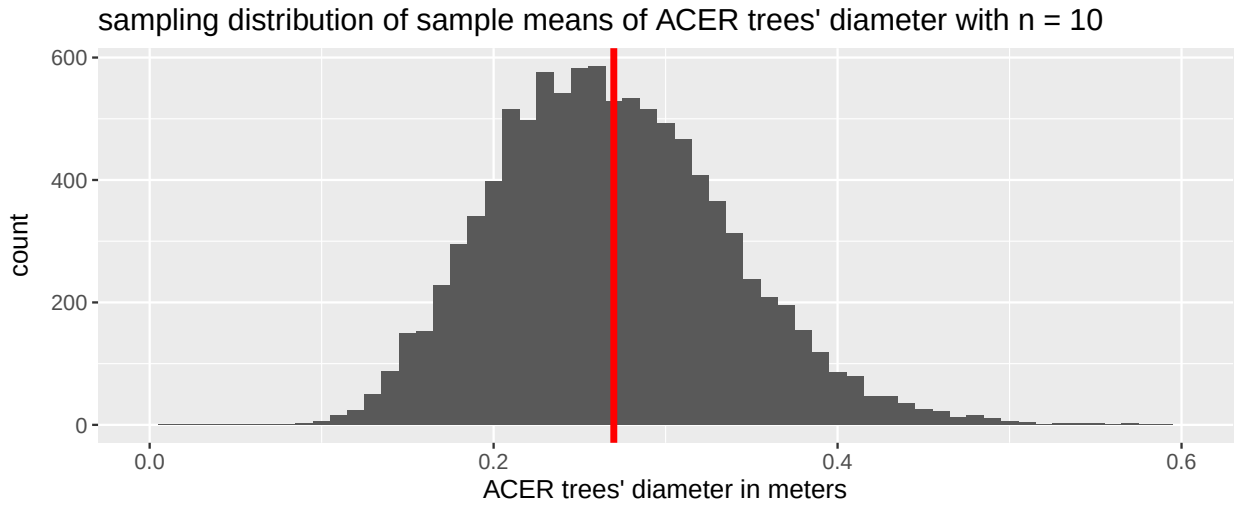
Ensure that your x and y-axes are human-readable. Moreover, include a title. ~~Assign your plot to an object called `single_acer_sample_plot`.~~

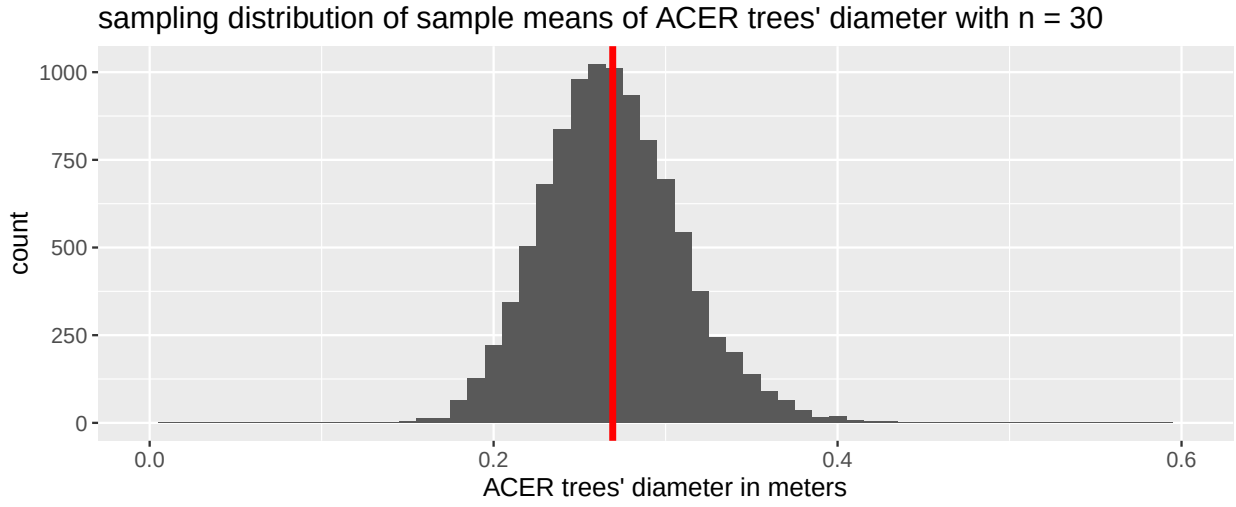
```
simulate_10k_10 <- acer_diameter |>
  simulate_10k(10)
simulate_10k_30 <- acer_diameter |>
  simulate_10k(30)
simulate_10k_100 <- acer_diameter |>
  simulate_10k(100)
```

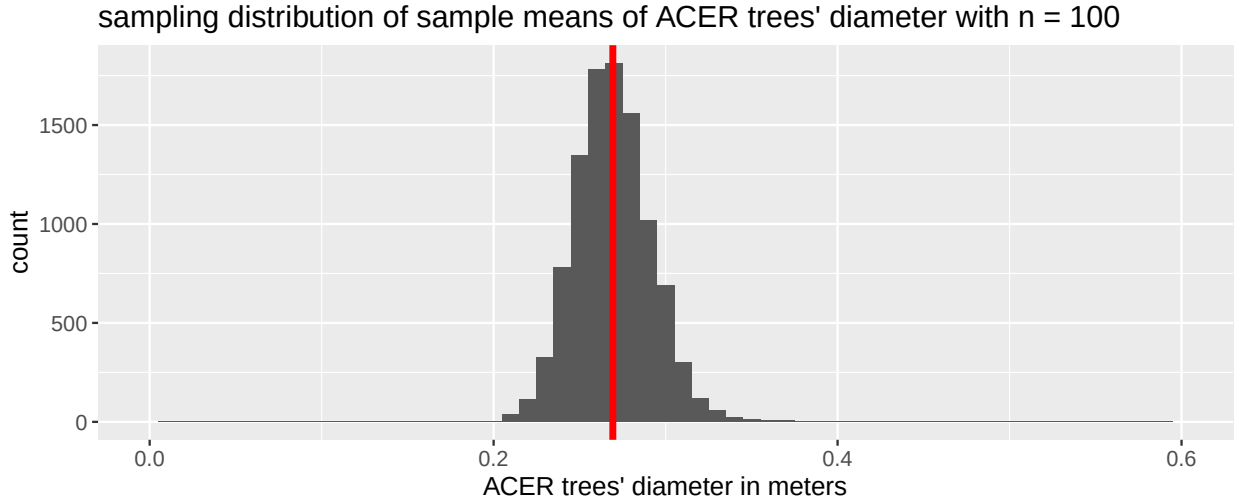
```
plot_sampling_dist_mean <- function(sampling_result, sample_size) {
  sampling_result |>
  ggplot(aes(x = mean)) +
  geom_histogram(binwidth = 0.01) +
  xlim(0, 0.6) +
  geom_vline(xintercept = mean(sampling_result$mean), color = "red", size = 1.5) +
  ggtitle(paste0("sampling distribution of sample means of ACER trees' diameter with n = ", sample_size
  xlab("ACER trees' diameter in meters") +
```

```
  theme(text = element_text(size = 12))
}

plot_sampling_dist_mean_10 <- simulate_10k_10 |>
  plot_sampling_dist_mean(10)
plot_sampling_dist_mean_30 <- simulate_10k_30 |>
  plot_sampling_dist_mean(30)
plot_sampling_dist_mean_100 <- simulate_10k_100 |>
  plot_sampling_dist_mean(100)
```

```
sampling_dist_by_n <- plot_grid(
  plot_sampling_dist_mean_10,
  plot_sampling_dist_mean_30,
  plot_sampling_dist_mean_100,
  align = "v",
  nrow = 3,
  ncol = 1
)

sampling_dist_by_n
```







Q3.3.

rubric={accuracy:1,reasoning:5}

Use the `knitr` `kable()` function to create a table to nicely display the sample size, mean, and standard error from `answer3_1`.

```
knitr::kable(answer3_1, "pipe")
```

n	mean	standard_error
10	0.2698865	0.0718193
30	0.2692608	0.0406214
100	0.2693146	0.0218887

In one paragraph, discuss the impact of changing sample size on the sampling distribution of the mean.

As the sample size increases, the standard error of the sampling distribution of the mean decreases, but the mean almost remains the same.

Exercise 4: Distribution of Bootstrap Sample Means

Q4.1.

rubric={autograde:4}

Take your single sample of $n = 100$ `ACER` tree diameters from **Exercise 2** stored in `answer2_3`, and call that your `one_sample` that you collected. Use that `one_sample` for bootstrapping using the corresponding `infer` package function to obtain 10,000 bootstrap samples. Then, calculate the sample means and standard deviations by bootstrap sample.

Store your results in `answer4_1`, which has to be a data frame with 10,000 rows and the following three columns:

- the bootstrap sample ID number (i.e., `replicate` from `rep_sample_n()`),
- the `mean` computed by bootstrap sample, and
- the `standard_deviation` computed by bootstrap sample.

Heads-up: Do not forget to seed your seed to `552`.

```
one_sample <- answer2_3
set.seed(552)
one_sample_bootstrap <- one_sample |>
  rep_sample_n(size = 100, replace = TRUE, reps = 10000) |>
  group_by(replicate) |>
  summarise(mean = mean(diameter), standard_deviation = sd(diameter))

answer4_1 <- one_sample_bootstrap
answer4_1
```

```
## # A tibble: 10,000 x 3
##    replicate  mean standard_deviation
##        <int> <dbl>              <dbl>
##  1         1 0.287              0.218
##  2         2 0.258              0.196
##  3         3 0.259              0.240
##  4         4 0.250              0.188
##  5         5 0.279              0.216
##  6         6 0.269              0.193
##  7         7 0.263              0.200
##  8         8 0.287              0.191
##  9         9 0.226              0.178
## 10        10 0.238              0.192
## # ... with 9,990 more rows
```

```
. = ottr::check("tests/Q4.1.R")
```

```
##
## All tests passed!
```

Q4.2.

rubric={autograde:2}

Calculate the mean and the standard deviation of the bootstrap distribution stored in `answer4_1`. Bind your results to vectors `bootstrap_mean` and `bootstrap_sd`

```
bootstrap_mean <- mean(one_sample_bootstrap$mean)
bootstrap_sd <- sd(one_sample_bootstrap$mean)
```

```
# YOUR CODE HERE

bootstrap_mean
```

```
## [1] 0.265338
```

```
bootstrap_sd
```

```
## [1] 0.02046713
```

```
. = ottr::check("tests/Q4.2.R")
```

```
##
## All tests passed!
```

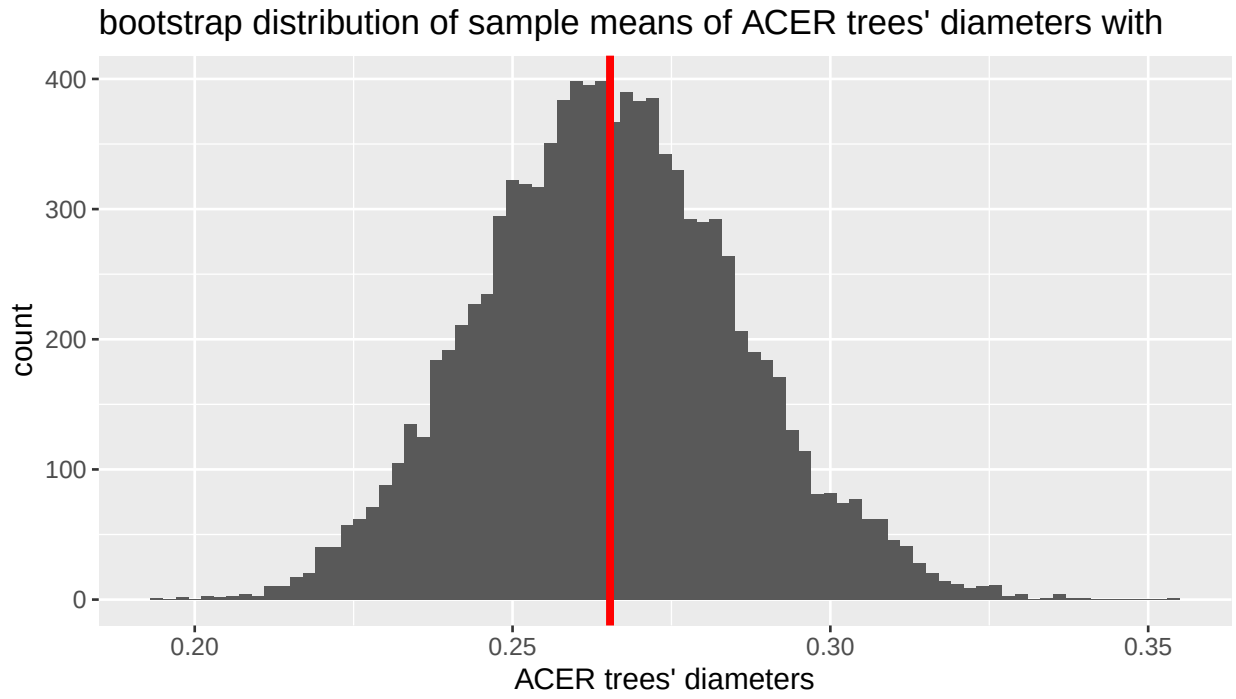
Q4.3.

rubric={autograde:2}

Visualize as a histogram the bootstrap distribution of sample means (stored in `answer4_1`) of `ACER` tree's diameters from the 10000 bootstrap samples of size $n = 100$.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to an object called `bootstrap_dist_100_plot`.

```
bootstrap_dist_100_plot <- one_sample_bootstrap |>
  ggplot(aes(x = mean)) +
  geom_histogram(binwidth = 0.002) +
  geom_vline(xintercept = mean(one_sample_bootstrap$mean), color = "red", size = 1.5) +
  # xlim(0, 1) +
  xlab("ACER trees' diameters") +
  ggtitle("bootstrap distribution of sample means of ACER trees' diameters with") +
  theme(text = element_text(size = 12))

bootstrap_dist_100_plot
```




```
. = ottr::check("tests/Q4.3.R")
```

```
##
## All tests passed!
```

Exercise 5: The Relationship Between the Sampling Distribution and the Bootstrapped Sampling Distribution

Q5.1.

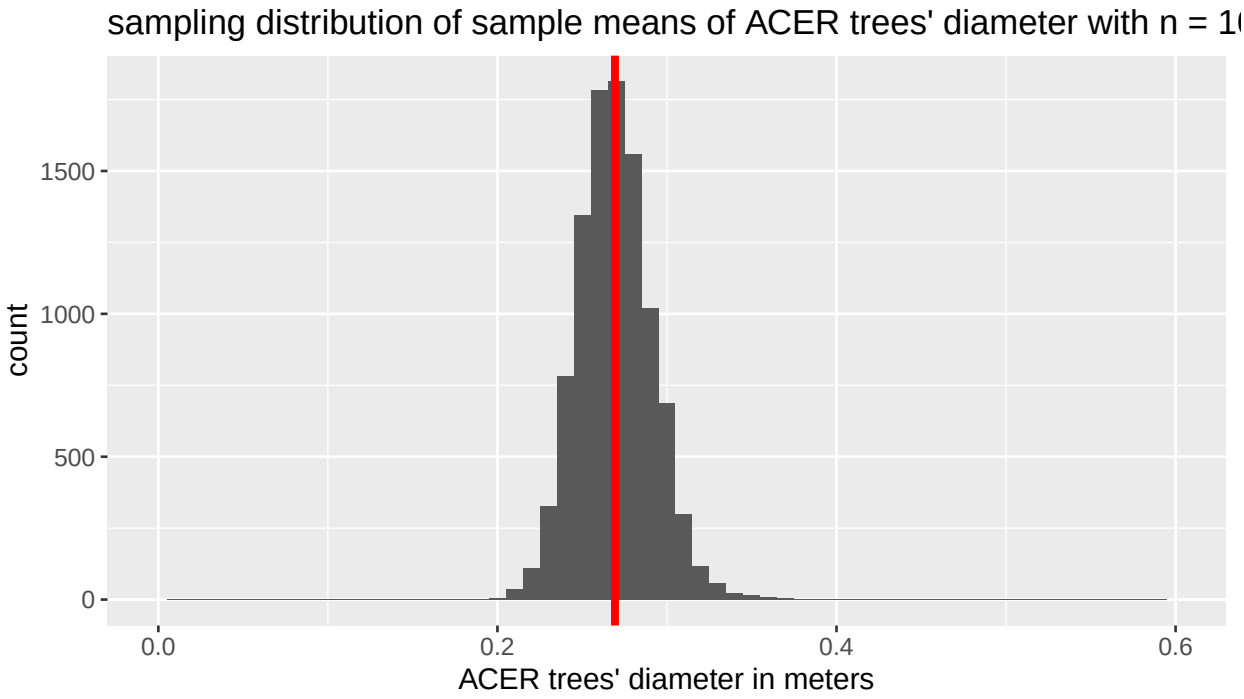
rubric={accuracy:1,viz:4}

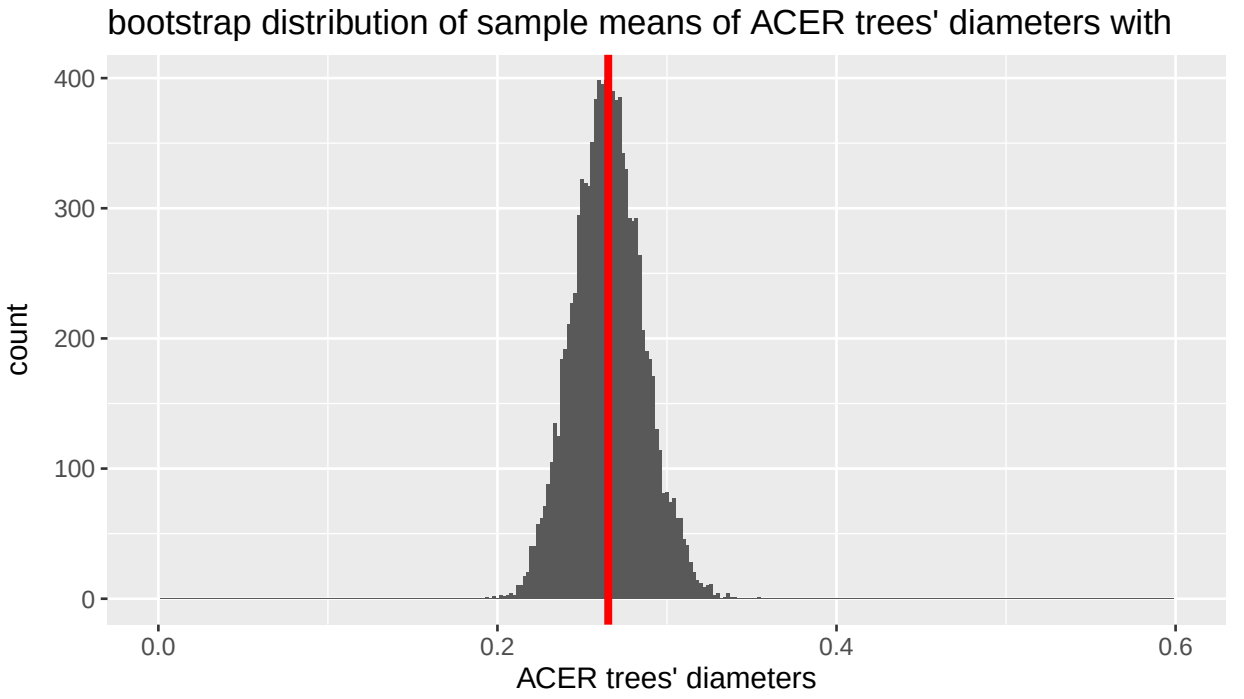
Visualize the distribution of your bootstrap sample mean and the sampling distribution of the mean (from **Exercise 1**) side-by-side. It would be useful to ensure that the x-axis have the same limits in both plots, and display them one-on-top of the other, to clearly see the differences and similarities.

```
compare_bootstrap_plot <- plot_grid(
  plot_sampling_dist_mean_100,
  bootstrap_dist_100_plot + xlim(0, 0.6),
  align = "v",
  nrow = 2
)
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
## Removed 2 rows containing missing values (geom_bar).
```

```
compare_bootstrap_plot
```





Calculate and report the means and the standard deviations of these two distributions (sampling distribution in `answer2_5` and bootstrap distribution in `answer4_1`) in a table using `knitr`'s `kable()` function.

```
sample_mean_stat <- answer2_5 |>
  summarise(mean_mean = mean(mean), sd_mean = sd(mean)) |>
  summarise(distribution_type = "sample mean",
         mean = mean_mean,
         standard_deviation = sd_mean,
  )
```

```r
bootstrap_sample_mean_stat <- answer4_1 |>
  summarise(mean_mean = mean(mean), sd_mean = sd(mean)) |>
  summarise(distribution_type = "bootstrap sample mean",
         mean = mean_mean,
         standard_deviation = sd_mean,
  )

compare_stat <- bind_rows(sample_mean_stat, bootstrap_sample_mean_stat)
kable(compare_stat, format = "pipe")
```

distribution_type	mean	standard_deviation
sample mean	0.2693146	0.0218887
bootstrap sample mean	0.2653380	0.0204671

Q5.2.

rubric={reasoning:10}

In two or three paragraphs, discuss the similarities and differences between these distributions. Finally, given that you generally will not know the sampling distribution nor have multiple samples to create the sampling distribution histogram, comment on how a bootstrap distribution might be helpful for estimating a population parameter.

These 2 distributions are almost identical. Both are good for estimating population parameters. However, in real life given that we don't have the sampling distribution, we can use a bootstrap distribution instead to estimate population parameters.

(Challenging) Exercise 6: DRY

Q6.1.

rubric={accuracy:5}

In **Exercise 3**, you might likely violate the DRY programming principle (**D**o not **R**epeat **Y**ourself). Write a function that takes the following parameters:

- `data`: the population data (e.g., the `ACER` tree),
- `col`: the column in the data frame (e.g., the diameters).
- `n`: a number vector specifying different sizes (e.g., `c(10,30,100)`).

This function should return a list with two elements:

1. `plot`: a `ggplot2` object (created using `plot_grid` that combines all the histograms into a panel plot).
2. `df`: a data frame containing the sample size `n`, means `mean` and standard error `standard_error` for each sample size given by the user. The output should look similar to **Q3.1.**

Note this **main function** could use **other auxiliary functions within itself**, if you think that is reasonable.

Include the corresponding docstrings for your function(s).

> **Heads-up:** Remember to `set.seed(552)` in your given sampling function.

```
compare_sample_size <- function(data, col, n) {

  # Remember to set.seed(552)
  plot <- NULL
  df <- NULL

  # YOUR CODE HERE
  return(list(plot, df))
}

# Run to test your function
compare_sample_size(answer2_1, diameter, c(10, 30, 100))
```

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

```
# This should return a data frame similar to answer3_1
compare_sample_size(answer2_1, diameter, c(10, 30, 100))[[2]]
```

```
## NULL
```

Q6.2.

rubric={accuracy:3}

Demonstrate that your function works by writing unit tests using testthat. Here are some example test cases that should be written:

1. The object being returned is a list with two elements, the first is a `cowplot` object (i.e., a grid of plots), and the second is a tibble or data frame.

2. The data frame should have 3 columns: `n`, `mean`, `standard_error` and the number of rows should be equal to the length of vector `sample_sizes`.

3. The standard errors for larger `n`'s should get smaller.

4. The standard errors should be smaller than the means.

```
# YOUR CODE HERE
```

Submission

Congratulations! You are done the lab!!! Do not forget to:

- Knit the assignment to generate `.pdf` file and push everything to your GitHub repo.
- Submit the `.Rmd` AND the `.pdf` files to Gradescope.